

Setup for the IT Service Vision Server

Introduction

Preliminary Checklist

Section 1, Getting Up and Running

The purpose of this section is to guide you through the steps that are necessary to collect representative performance data at your site, create a test IT Service Vision PDB, process the data into it, reduce the data in the PDB, and analyze and report on the data.

Section 1, Task 1: Start the IT Service Vision Server Software

Section 1, Task 2: Start Your Data Collector Software

Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data

Section 2, Making the Transition to Production

Section 1 got you started with data collection, a test PDB, and some simple report definitions. This section describes how to finish customizing the data collection, PDB, and report definitions to the specific requirements at your site. It also describes how to put the customized versions into production.

Section 2, Task 1: Customize and Verify Your Test PDB

Section 2, Task 2: Create Your Production PDB

Section 2, Task 3: Customize Your Report Definitions

Section 2, Task 4: Set Up Your Production Job

Section 2, Task 5: Prepare for Use of the Client

Section 2, Task 6: Wrap Up

Section 3, Advanced Setup

Section 3, Advanced Setup

Platform - Specific Appendices

Open Systems (UNIX)

Windows NT

Open Systems and Windows NT Appendix 1: Tables and Variables Naming Convention

Open Systems and Windows NT Appendix 2: Tips for Reporting

Open Systems and Windows NT Appendix 3: Printing Graphs

Open Systems and Windows NT Appendix 4: MIB to Dictionary Compiler

Collector - Specific Appendices

Open Systems | Cabletron SPECTRUM

Cabletron SPECTRUM Appendix 1: Starting the processd Daemon

Cabletron SPECTRUM Appendix 2: Alternate Data-Logging Method

[Cabletron SPECTRUM Appendix 3: SPECTRUM Specific Tips for Reporting](#)
[Cabletron SPECTRUM Appendix 4: Defining IT Service Vision Tables for SPECTRUM Data](#)
[Cabletron SPECTRUM Appendix 5: Editing Model Types for SPECTRUM](#)
[Cabletron SPECTRUM Appendix 6: Mapping SPECTRUM Data Types to an IT Service Vision PDB](#)

Open Systems | OpenView and NetView

[OpenView and NetView Appendix 1: Recommended Metrics](#)
[OpenView and NetView Appendix 2: Alternate Data-Logging Method](#)
[OpenView and NetView Appendix 3: OVW-Specific Tips for Reporting](#)

Open Systems | SunNet Manager and Enterprise Manager

[SunNet Manager and Enterprise Manager Appendix 1: Mapping SunNet Manager Metrics to IT Service Vision Tables and Variables](#)
[SunNet Manager and Enterprise Manager Appendix 2: Defining Tables from Your Own Schema Files](#)

Introduction

This documentation contains setup instructions and information for the IT Service Vision server for Open Systems and Windows NT supplied collectors.

Some of the instructions and information apply across all platforms, all collector facilities, and all data sources and collectors. Some of the instructions and information apply only to a specific platform, facility, data source, or collector. The instructions and information that differ are identified with icons.

These labels and icons will represent your platform for the rest of the document.











For many data sources, IT Service Vision supplies specific table definitions and staging software for using the data with IT Service Vision. The collectors whose logs are supported this way are called *supplied (table definitions and staging code) collectors*. For each of these supplied collectors, IT Service Vision provides:

- **SAS code that stages the data (as staging data sets or DATA step views)**
- **one or more table definitions that you can add to your performance data warehouse (PDB) and several ways to add the table definition(s) to your PDB**
- **a macro that invokes IT Service Vision**
- **a macro that processes the staged data into the tables in your PDB**
- **a macro that reduces the data in the PDB**
- **macros that report on the data in the PDB**
- **utility macros to manage the PDB.**


Note: Some of these macros are used by more than one collector.

On the Open Systems platforms (IBM® AIX®, Sun Solaris, and HP-UX), IT Service Vision supplies table definitions and staging code for data from the following data collectors:

Note: These labels and icons will represent your collector for this rest of the document. Also in the [IT Service Vision Showroom](#), there may be additional documentation on these collectors.

Label	Icon	Collector
ACC		UNIX Accounting
CSi		SPECTRUM by Cabletron Systems, Inc.
MWA		Performance Collection Software and MeasureWare by Hewlett-Packard Company
NV		NetView for AIX by IBM
OV		OpenView Network Node Manager by Hewlett-Packard Company
PRX		PROBE/Net by Landmark Systems
SNM		SunNet Manager by Sun Microsystems, Inc.
TKR		TRAKKER by Concord Communications, Inc.



Note: In this document the label **OVNV** and the icon  represent both OpenView and NetView products.



Note: In the IT Service Vision Showroom, there also are examples of working with data from other data sources. Some examples are:

Label	Collector
NT SMF	NT SMF by Demand Technology Software
ROLM	Rolm PBX Traffic
R/3	R/3 by SAP AG
Web	Web Server Log Data
DECps	Polycenter Performance Solution by Digital Equipment Corporation
SNMP	SNMP mib-II data
vmstat	UNIX vmstat

Additionally, the [IT Service Vision Showroom](#) contains information on how to web-enable reports on data from any collector.

On Windows NT, IT Service Vision supplies table definitions and staging code for data from the following data collectors:

Note: These labels and icons will represent your collector for the rest of this document. Also, in the [IT Service Vision Showroom](#), there may be additional documentation on these collectors.

Label	Icon	Collector
MWA		MeasureWare by Hewlett-Packard Company
OV		OpenView Network Node Manager by Hewlett-Packard

Note: In the [IT Service Vision Showroom](#), there also are examples of working with data from other data sources . Some examples are:

Label	Collector
NT SMF	NTSMF by Demand Technology Software
ROLM	Rolm PBX Traffic
R/3	R/3 by SAP AG
Web	Web Server Logs

Additionally, the [IT Service Vision Showroom](#) contains information on how to web-enable reports on data from any collector.

Preliminary Checklist

This checklist describes the information that you may need to gather and verify before you set up IT Service Vision. This checklist also has advanced topics that you may want to make use of after setup.

Suggested Reading

- In this document, you will see references to SAS windows, pull-down menus, and command lines. For more information on using these in IT Service Vision, see [Shared Appendix 2: Navigating SAS Windows](#).

If you are new to performance evaluation and want to do some background reading, we suggest the following books:

- *Operating System Concepts, Fourth Edition*. By Abraham Silberschatz and Peter Galvin. Addison-Wesley Publishing Company, Inc. 1994. ISBN 0-201-50480-4. This book is a good introduction to the purposes and internals of operating systems in general.
- *Merrill's Expanded Guide to Computer Performance Evaluation Using the SAS System*. By H.W. "Barry" Merrill. SAS Institute Inc. 1984. ISBN 0-917382-54-4. Focus on concepts, relationships, and analysis techniques. See Chapters 1, 2, and 7-30.

You may also want to read the following papers. They are platform-specific but general enough to show some of the concepts in use. They are in materials from the Computer Measurement Group (CMG).

Computer Measurement Group, Inc.
P.O. Box 1124
151 Fries Mill Rd.
Suite 104
Turnersville, NJ 08012
USA
Phone Number (800) 4 FOR CMG or (609) 401-1700
FAX Number (609) 401-1708
<http://www.cmg.org>

- "Introduction to Performance Management: What to Measure, What to Report." By Stephen Samson. CMG 94 Proceedings. Volume 1.
- "Getting Started in Performance Tuning and Capacity Planning." By Neil Ervin. CMG 94 Proceedings. Volume 2.
- "An Introduction to Time Series Forecasting for CPE." By Brian Bennett. CMG Transactions. Summer 1994.
- "Network Analysis Case Study." By Barbara Pendergrass and Robert N. Bonham. CMG 94 Proceedings. Volume 2.
- "Distributed System Analysis Case Study." By Barbara H. Pendergrass and Robert N. Bonham. CMG 95 Proceedings. Volume 2.

Are you familiar with the SAS System and the SAS language? The IT Service Vision graphical user interface empowers you to access, manage, analyze, and present performance evaluation information without knowing SAS software. But you can extend the capabilities of the product by using the SAS language and other SAS products. If you want to learn more about the SAS software and language, a good place to start is *SAS Language and Procedures: Introduction, Version 6 First Edition* and/or *The Little SAS Book: A Primer* by Lora D. Delwiche and Susan J. Slaughter, Cary, NC: SAS Institute Inc., 1995, ISBN 1-55544-215-3.

For an Overview of IT Service Vision, see [Shared Appendix 1: Overview](#).



Because Open Systems software comes from so many vendors, the best sources of general purpose information are books. The following are recommended:

- *Applications for Distributed Systems and Network Management*. By Kornel Terplan and Jill Huntington-Lee. Van Nostrand Reinhold. 1994. ISBN 0-442-01873-8.
- *System Performance Tuning*. By Mike Loukides. O'Reilly and Associates. 1991. ISBN 0-937175-60-9.
- *UNIX System Administrator's Handbook*, Second Edition. By Evi Nemeth, Garth Snyder, and Scott Seebass. Prentice Hall. 1995. ISBN 0-131510-51-7.
- *Managing NFS and NIS*. By Hal Stern. O'Reilly and Associates. 1992. ISBN 0-937175-75-7.

The Landmark Systems Corporation sells a book *UNIX as a Second Language*. The book explains UNIX performance tuning in terms of MVS performance tuning. For more about the book, contact Landmark Systems Corporation, 8000 Towers Crescent Drive, Vienna, VA 22182-2700, Phone: 1-800-775-5675.

Consult the documentation of your collector software for terminology and other suggested reading. Also, select the appropriate icon for your collector software below for additional information.



Abbreviations and Terms for OpenView and NetView:

- *NVAIX* refers to all IBM NetView for AIX components
- *OVW* refers to HP OpenView Windows (a graphical user interface)
- *NNM* refers to HP OpenView Network Node Manager (an application that runs under OVW)
- *HPOV* refers to all HP OpenView products:
 - OVW
 - NNM
 - the collection of background processes (daemons) that these products use.
- *SNMP host or SNMP management host* is a host machine running either HP Open View or IBM NetView for AIX
- *SNMP management software* is either HP Open View software or IBM NetView for AIX software.

For detailed information on HP OpenView Network Node Manager or IBM NetView for AIX, please refer to the following Hewlett Packard or IBM publications:

- *HP OpenView Network Node Manager User's Guide*
- *HP OpenView Network Node Manager Administrator's Reference*
- *HP OpenView Windows User's Guide*
- *IBM NetView for AIX User's Guide for Beginners*
- *IBM NetView for AIX Administrator's Guide*
- *IBM NetView for AIX Problem Determination.*



Abbreviation and Terms for Cabletron SPECTRUM

Caution: The SPECTRUM examples and definitions are meant as a guide to setting up SPECTRUM. The examples and definitions were created using SPECTRUM Release 4.0 and work with that release. In cases where there is a discrepancy between the examples and definitions in this document and those in Cabletron SPECTRUM documents, the information from Cabletron takes precedence

- A *landscape* is the set of devices managed by a specific server. That is, a landscape is the server plus the devices managed by that server.
- A *SPECTRUM database* contains general information about the types of devices in a landscape and also contains current data collected about the devices in a landscape. Generally, *current* refers to the last few cycles of data collection.

SPECTRUM archives collect data about devices in a statistics data base. Data are exported from the statistics data base to SAS staging data sets. The data in the data sets are then processed into your PDB.

- A *model type* is a set of similar devices. One model type might be HP workstations, another might be Cabletron Systems MRXI Repeaters, and so on. Each model type has a name, and model type names are case sensitive.
- A *model type handle* is a hex number that uniquely identifies a particular model type. Model type handles are assigned by SPECTRUM and, thus, are the same in every SPECTRUM landscape. For instance, HP workstations have a model type name of Host_HP, which corresponds to a model type handle of 0x1160008.

Note: A table in an IT Service Vision performance data warehouse (PDB) contains the data from a given model type. The IT Service Vision table name is the same as the model type name (or shortened if the full name of the model type is too long).

- A *model* is a particular device managed by a server. Thus, a model is an instance of a model type. A SPECTRUM database typically has data about more than one model per model type.
- A *model handle* is a hex number that uniquely identifies that particular model (device) in a SPECTRUM database. A model handle is generated by SPECTRUM when a device is added to the landscape, and the model handle is unique within that landscape. For example, a particular Cabletron MRXI repeater could have model handle 0x04023a1 on one server, and no other device on that server would have that same model handle. And if that Cabletron MRXI repeater were also managed by another server, its model handle would be 0x04023a1 in the first SPECTRUM database and could be 0x04023a1 or, say, 0x23001b in the other SPECTRUM database.

- An *attribute* is the name of a variable (metric) that is defined for a model type and whose values are assigned or measured for the models of that model type. The attributes for each model type are controlled by that model type's Management Module. For example, some of the attributes for Cabletron MRXI repeaters are Model_handle (model id), Model_type_handle (model type id), and IP_In_Discards (the number of IP layer packets discarded). Thus, a SPECTRUM database generally has data about more than one attribute per model and, for all models of the same model type, has data on the same attributes.
- An *attribute ID* (attrid) is a hex number that uniquely identifies that particular attribute. Attribute IDs are assigned by SPECTRUM. For example, Model_type_handle is attribute ID 0x10001, IP_In_Discards is attribute ID 0x1009d, and so on.
Note: The variable names in the IT Service Vision performance data warehouse are the same as the SPECTRUM attribute names (or shortened if the attribute names are too long).

In this document, the following terms refer to the Spectrum software:

- *SpectroSERVER* is the name of the SPECTRUM server software. It consists of the knowledge base server and a set of background processes (daemons).
- *SPECTRUM_home* (also called *SPECTRUM_path* or *SPECTRUM_top*) is the directory in which the SPECTRUM server software is installed. A typical value for SPECTRUM_path is /usr/spectrum .
- *SpectroGRAPH* is the name of the SPECTRUM client software.
- *SPECTRUM Data Export* (SDE) software is a tool that exports SPECTRUM data into SAS data sets (or ASCII files).
- *SPECTRUM* refers to the following:
 - SpectroSERVER
 - SpectroGRAPH
 - SPECTRUM Data Export (SDE)
 - other applications and tools that these three products use.

For detailed information on SPECTRUM, please refer to the following Cabletron Systems publications:

- *SPECTRUM System Installation Guide*
- *SPECTRUM System Administrator's Guide*
- *SPECTRUM System User's Guide*
- *SPECTRUM Data Export User's Guide*
- *SPECTRUM Using Data Export and Reports with the Distributed Data Manager .*



- *Microsoft Windows NT® Workstation Resource Kit*. By Microsoft Press. 1996. ISBN 1-57231-343-9. See Part III, *Optimizing Windows NT Workstation*.
- See the white paper written by Demand Technology: *Windows NT: Can it be tuned* on the World Wide Web at <http://www.demandtech.com/>
- See the white paper written by Compaq: *Performance Tuning Your Windows NT Web Server* on the World Wide Web at <http://www.compaq.com/support/techpubs/whitepapers/457a1096.html>.
- See the white papers written by NCR on the World Wide Web at <http://www.ncr.com/support/nt/ntrefs.htm#WHITEPAP>.

Consult the documentation of your collector software for terminology and other suggested reading. Also, select the icon below for additional information.



Abbreviations and Terms for OpenView:

- *NVAIX* refers to all IBM NetView for AIX components
- *OVW* refers to HP OpenView Windows (a graphical user interface)
- *NNM* refers to HP OpenView Network Node Manager (an application that runs under OVW)
- *HPOV* refers to all HP OpenView products:
 - OVW
 - NNM
 - the collection of background processes (daemons) that these products use.
- *SNMP host or SNMP management host* is a host machine running either HP Open View or IBM NetView for AIX
- *SNMP management software* is either HP Open View software or IBM NetView for AIX software.

For detailed information on HP OpenView Network Node Manager or IBM NetView for AIX, please refer to the following Hewlett Packard or IBM publications:

- *HP OpenView Network Node Manager User's Guide*
- *HP OpenView Network Node Manager Administrator's Reference*
- *HP OpenView Windows User's Guide*
- *IBM NetView for AIX User's Guide for Beginners*

- *IBM NetView for AIX Administrator's Guide*
- *IBM NetView for AIX Problem Determination.*

Section 1, Getting Up and Running

The purpose of this section is to guide you through the steps that are necessary to collect representative performance data at your site, create a test IT Service Vision PDB, process the data into it, reduce the data in the PDB, and analyze and report on the data.

Section 1, Task 1: Start the IT Service Vision Server Software

This task walks you through the basic site-level customization of the IT Service Vision server software. (Customization of the IT Service Vision client software is covered in a later task.)

Prerequisites

1. You know the work shift schedule and holiday schedule at your site.

You need a schedule of the work shifts that are important for your performance reporting. You can define as many as six shifts with the server's interactive interface. If you want to assign more than six work shifts, see *How to Assign More than Six Shifts* in the portable section of Issue 6 in the newsletter. For the newsletter, follow this path:

Online Help -> Documentation Contents -> Internet Links -> WWW Home Page

Work shifts are defined by hour within day. For example, the default schedule in IT Service Vision has two shifts: shift 1 is Monday through Friday from 8 a.m. to 4:59 p.m. (08:00 - 16:59), and shift 2 is the remainder of the week.

You also need a list of the dates that are holidays at your site. A list for the current year is good. A list for the current year and the next year is even better. (There is a default holiday list, although it will probably not match the holidays used at any particular site.)

You may want to check with other performance areas at your site so that all areas are using the same schedules and holidays so that all reports are consistent.

Actions

1. Invoke the IT Service Vision server interface.

From your operating system prompt use one of these methods:



You can start IT Service Vision in one step or two. Use whichever method works best for you.

a. To start IT Service Vision directly, issue this command:

The command is:

```
itsv -sasroot sasroot
```

where `itsv` is the name of a shell script in your IT Service Vision `misc` directory and `sasroot` is the location at which SAS is installed.

For example:

```
cd !cproot/misc
./itsv -sasroot sasroot
```

or

```
!cproot/misc/itsv -sasroot sasroot
```

b. To start SAS and then start IT Service Vision, issue these commands:

- i. Invoke SAS by issuing this command at your shell prompt

```
sas -dms &
```

(or the appropriate site command if your site customized the command for starting SAS)

- ii. From within SAS, do one of the following

- Follow this path

Globals -> Options -> Command -> type CPE -> OK

- Or, in the body of the PROGRAM EDITOR window, type

```
%cpstart( );
```

and then follow this path

Locals -> Submit

- iii. After IT Service Vision server interface displays the IT Service Vision splash screen. Select **OK** to Continue. Now, IT Service Vision displays the IT Service Vision main window.



You can start IT Service Vision in one step or in two steps. Use whichever method works best for you.

a. To start IT Service Vision directly, issue this command from from Start->Run... (or from a DOS window):

The command is:

```
itsv
```

where `itsv` refers to the file `ITSV.BAT` in your IT Service Vision `misc` directory. This file must be edited to point to the location of SAS on your system.

b. To start SAS and then start IT Service Vision, issue these commands:

- i. Invoke SAS using the appropriate command for your system (see the installation documentation for the SAS System or contact your SAS system administrator).

- ii. From within SAS, do one of the following

- Follow this path

Globals -> Options -> Command -> type CPE -> OK

- Or, in the body of the PROGRAM EDITOR window, type

```
%cpstart();
```

and then follow this path

Locals -> Submit

- iii. After IT Service Vision server interface displays the IT Service Vision splash screen. Select **OK** to Continue. Now, IT Service Vision displays the IT Service Vision main window.

If you installed ITSV2 in a temporary directory (that is, not in the same directory that the SAS System is in), you need to specify the location of IT Service Vision when you start it. To do this, enter the **cpe** command from the SAS command line, specifying the temporary IT Service Vision directory for the ROOT location:

```
cpe root='temporary-directory' pdb='temporary-directory/pdb-pbx'
```

You can enter the **cpe** command in the SAS toolbox command area or after the *Command*====> prompt (obtained by clicking on **Globals > Options > Command Line**).

You can also start IT Service Vision from within SAS by submitting this statement from the PROGRAM EDITOR window:

```
%CPSTART( root='temporary-directory/',
          pdb='temporary-directory/pdb-pbx' );
```

To start IT Service Vision from your Windows desktop, click on **Start > Programs > IT Service Vision > Start IT Service Vision**. If you get the message:

Out of environment space

when you start IT Service Vision from your Windows desktop, you need to increase the amount of space allocated by DOS for environment variables for this process. From an Explorer window, locate the file **sasmisc\itsv.bat** in the directory in which you installed IT Service Vision. Click the right mouse button on the file and select **Properties > Program**. Click to check the box next to "Close on exit". Select the **Memory** tab and click on the down arrow next to "Initial environment". Select a large number (for example, 4096). Click on **OK** to apply the changes and follow the path described above.

2. Obtain write access to SITELIB.

From the IT Service Vision main window, select

Administration -> Site Options -> General -> select the down arrow on the Sitelib Access field -> WRITE -> OK

Note: Both the CPE command and the %cpstart macro invoke IT Service Vision.

If, in other tasks, you want to invoke IT Service Vision with parameters, create a file containing the **%CPSTART** macro with the desired parameters and start SAS by pointing to the file as follows (see the SAS System documentation for more information on using the **-autoexec** option):

```
sas -autoexec mypgm.sas
```

or start SAS and type statements in the body of the PROGRAM EDITOR window. Examples of these statements are:

- This statement invokes the IT Service Vision server with a pdb named pdb-test as the active PDB, read access to

the PDB, and write access to SITELIB:

```
%CPSTART( pdb='/tmp/pdb-test', siteacc=write );
```

- This statement invokes the IT Service Vision server with a pdb named pdb-test as the active PDB, write access to the PDB, and read access to SITELIB:

```
%CPSTART( pdb='/tmp/pdb-test', access=write );
```

- This statement invokes the IT Service Vision server with a pdb named pdb-test as the active PDB and write access to the PDB and write access to SITELIB

```
%CPSTART( pdb='/tmp/pdb-test', access=write, siteacc=write );
```

Then, from your PROGRAM EDITOR window, select **Locals** and **Submit**.

3. Adjust the default work shift schedule and holiday schedule.

a. Request read/write (update) access to SITELIB.

Administration -> Site Options -> click on down arrow to the right of the Sitelib Access field-> WRITE -> OK

b. Display the Define Work Shifts window.

You can use work shifts and holidays to subset data for reports. Follow this path:

Administration -> Define Work Shifts

IT Service Vision displays the Define Work Shifts window. In the upper right corner of the window is the Define at Which Level? subwindow. Select Site. (When new PDBs are created, they automatically take their defaults from the Site Defaults. You can also manually apply the Site Defaults to an existing PDB.)

i. Define and edit the work shifts schedule.

- In the lower left corner of the window is the Shift Definition subwindow. It lists the default shift codes and their descriptions (meanings).
- If the codes and descriptions do not provide the distinctions that you want to make among shifts, overwrite the codes and meanings with ones that are suitable for your site.
- In the upper left corner of the window is the Shifts by Day of Week subwindow. The title is a day-of-week. The body of the window contains a list, for that day, in which each line has an hour of the day (in 24-hour clock time; for instance, 14:00 means 2:00 pm - 2:59 pm) and the shift code to which it is assigned.
- Overwrite the displayed shift codes with the shift codes that apply at your site. To overwrite a shift code, click on it and then enter the new shift code for that hour. To see the shift codes for the rest of the day, use the vertical scroll bar.

Select the next day, by moving the scroll bar in the Shifts by Day of Week subwindows, and set the shift codes for that day. Repeat until you have set the shift codes for every day of the week.

ii. Define and edit the holidays schedule.

On the right in the window is the **Define Holidays** button. Select it. IT Service Vision displays a series of lines, each of which has a date, the status of the date (**Active** means the date is a holiday at your site), and a description of the holiday.

Modify the default list so that it correctly represents the holiday list at your site.

- To change the status or description of a date already on the list, select the date and then ***right mouse click*** and **Edit**. Change the status and/or description, and then select **OK**.
- To add a date to the list, select **Actions** and **Add Holiday**. Fill in the date, select **Active**, fill in the description, and select **OK**.
- To delete a date from the list, select the date, select ***right mouse click*** and **Delete**, and confirm with **OK**.
- Select **File** and **Close** to return to the Define Work Shifts window. Select **OK** to exit this window.

4. **Exit to your operating system prompt.**

Follow this path:

File -> Exit... -> Exit IT Service Vision and SAS -> OK

Section 1, Task 2: Start Your Data Collection Software

This task walks you through checking the installation of your logging and collector software and then starting the logging and collection.

Prerequisites

In this document, the host on which your collector software is installed is called the *collector host* or the *management host*. You can use your workstation as the collector/management host. However, typically the collector/management host is a different machine from your workstation.

1. Verify that the logging and collector software is installed appropriately.

Verify that your *logging and collector* software is installed properly according to the vendor's documentation.

For performance reasons, the *collector host* running the software that actually logs the performance data will generally be a different host from your workstation and from the host upon which you do your data processing and analysis with IT Service Vision.

Actions

1. Start the logging and collector software, if it is not already running.



Start process accounting to a disk file with the `accton` command on each system you want to monitor. Though the exact syntax of the `accton` command varies slightly from one UNIX operating system to the other, it is generally the case that process accounting is initiated when root or another authorized userid issues the `accton` command. As processes end, information on the resources they consumed is logged by the operating system. See the man file for the UNIX system in question for details on running the `accton` command.



a. Start SpectroSERVER.

With SPECTRUM Release 4.0, the recommended method for starting the server is by using the SPECTRUM Control Panel. Using the SPECTRUM Control Panel insures that the environment is properly set up and that the supporting daemons, DAS and ArchMgr, are running. For versions prior to 4.0, please refer to your *SPECTRUM System Administrator's Guide* for the proper method for starting SpectroSERVER.

i. **Check that the SPECTRUM processd daemon is started.**

Issue this command, at your workstation:

```
spectrum
```

You can type this command from any directory.

SPECTRUM displays the Select Host Machine window (and the Control Panel window). Display of the Select Host Machine window indicates that the processd daemon is started. On the Select Host Machine window, select the name of the SPECTRUM management host or select **DEFAULT**. Select **OK**.

SPECTRUM closes the Select Host Machine window.

Note: If the Select Host Machine window does not display, and instead a window displays with a message similar to this (where *machine name* is the SPECTRUM management host name):

```
Cannot connect to the processd at this location:
```

```
machine name
```

Then the SPECTRUM processd daemon on the management machine is not started. See [Cabletron SPECTRUM Appendix 1: Starting the processd Daemon](#) for instructions on starting processd. After you start processd, repeat this action.

ii. **Check that SpectroSERVER is running.**

When the Select Host Machine window closes, the Control Panel window is left open. In the Message subwindow, SPECTRUM displays

```
SpectroSERVER is now ready...
```

and in the Status field SPECTRUM displays

```
RUNNING
```

This combination of values indicates that SpectroSERVER is running.

Note: If one or both values are *not* as described, select **Start SERVER** on the Control Panel window. In the Status field, SPECTRUM displays STARTING and then displays RUNNING. In the Message subwindow, SPECTRUM displays

```
SpectroSERVER is now ready...
```

When this combination of values is present, go on to the next action.

b. **Start SpectroGRAPH.**

On the Control Panel window, select **SpectroGRAPH**. The window named Select Server To Connect To... is displayed. Select the name of the SPECTRUM management machine or select **DEFAULT**. Then select **OK**. SPECTRUM closes the Select Server To Connect To ... window. After a pause, SPECTRUM displays and closes the *eye* window and then displays the Topology View window and perhaps other View windows (Alarm View or Geographic View or Hierarchy View).

Note: If, instead of the *eye* window and View window, SPECTRUM displays a window with a message similar to this one

Error: No SpectroSERVER to talk to!

then SpectroSERVER is not running on the selected host. Select **Close** on this window. Repeat this action and select the correct host name (the one that you selected for SpectroSERVER).

In this document we suggest you export and examine the data one model type at a time. Select a model type. We suggest using type HubCSIMRXi, a common Cabletron Systems MRXI Repeater. If you do not have models of this type, whenever you see references in this document to HubCSIMRXi, substitute another model type that you do have, preferably a common model type. To determine the model type for most models, follow this path from the Topology View window:

Select the Model -> View -> Icon Subviews -> Model Information

The Model Type field contains the model type name that SPECTRUM uses. To close this window and go back to the Topology View window, select **File** and **Close**.

If there is no Model Information for that model type, check the SPECTRUM document *Management Module Guide* for that device.

Perform all the steps in this task from within SpectroGRAPH. If the Topology view window is partially or fully overlaid, make it the top window so that it is fully visible.

c. Select the first node of type HubCSIMRXi from which you want to collect data.

You may need to select down to a lower layer of the map to locate the device of interest.

The "Find" view can be very useful for locating a particular model or all models of a particular model type.

i. Locate all nodes of type HubCSIMRXi.

Select...

View -> New View -> Find...

When the Find Attribute Selection window displays, select

Model-Type Name -> OK

Type HubCSIMRXi (remember that model type names are case sensitive) and select **OK**, in the Find Models with Model-Type Name field.

SPECTRUM displays a list of all models of this model type.

ii. Select a node from the list.

Double-click on the node name.

The icon representing that node displays above the list.

d. Begin polling and logging data for the device.

i. Select the appropriate icon subview.

When an icon representing a HubCSIMRXi is visible, follow this path

View -> Icon Subviews -> Model Information

to select the icon subview that controls polling and logging of attributes for the HubCSIMRXi devices. For model types other than HubCSIMRXi, use the *Management Module Guide* to determine which icon subview controls polling and logging of attributes for your device.

- **Set the Poll Interval field to 60 to collect data every minute.**

When changing the value for this field, you may be asked to confirm the changes. Select **OK** to keep the changes.

- **Set the Log Ratio field to 10 to log data every ten minutes.**
- **Select File and Close to exit from the Model Information icon subview.**

SPECTRUM begins the polling and logging.

ii. Verify what is being logged.

Follow this path:

View -> Icon Subviews -> Model Information

and look at the LOGGED list box. The attributes that are now being logged display in this section.

Note: If the attributes listed do not describe the performance of the device sufficiently for your purposes, you may need to edit the model type definition so that it logs other attributes in addition to the ones shown. See [Cabletron SPECTRUM Appendix 5: Editing Model Types for SPECTRUM](#) for information on adding attributes to be logged.

iii. Repeat these steps for the remaining devices of type HubCSIMRXi for which you want to collect data.

Note: If you find this interactive method too time-consuming, see [Cabletron SPECTRUM Appendix 2: Alternate Data-Logging Method](#).

Note: You can exit SpectroGRAPH now (by selecting **File** and **Exit**) without affecting data polling and logging.

e. Edit the export script.

The script that executes the actual data export process must be configured for your site.

i. Edit this file in any ASCII editor.

`SPECTRUM_home/SG-Tools/dtxscript`

ii. Delete the comment delimiters (#s) that are at the beginning of these lines:

```
# SASROOT=  
# export SASROOT
```

iii. Specify the location of base SAS software.

SASROOT is the path to base SAS the lines to software. Add the SASROOT path to the line that now begins SASROOT=. For example, change it to this:

```
SASROOT=/local/sas
export SASROOT
```

iv. **Save the edited version and exit from the editor.**

Note: If you want to know more about `dtxscript`, see the document *SPECTRUM Data Export User's Guide*.

f. **Export the data using SDE and the export script.**

SDE uses Export Definition Descriptions, which are templates that control the export process. Because this is your first time using SDE, you will need to create one of these descriptions.

i. **Start SDE.**

Select the following from the main SpectroGRAPH window:

File -> Export Data...

ii. **Create and save a description for exporting the data.**

- **Set the `Landscapes...` field to the name of the SPECTRUM management host.**
- **Set the `Output Format` field to `SAS`.**
- **Select `Statistics` in the middle third of the SDE window.**

SPECTRUM displays the SDE: Statistics Filter window.

i. **Double-click on the `HubCSIMRXi` entry in the `Models/Model Types` field in the Statistics Filter window.**

ii. **Select all the models of interest by highlighting their model names.**

Hint: To select all models for a particular model type, such as `HubCSIMRXi`, select the model type name. SPECTRUM will highlight all models belonging to that model type.

iii. **Select the attributes that you want to keep in your PDB.**

Select Attributes from the list of all attributes that are listed in the `Common Attributes` part of the window.

Later, you will want to review the attributes that you select here, in order to check that they match the variables kept in your PDB.

iv. **Select OK.**

- **Specify frequency of exporting data.**

To specify that the data that were collected on the previous day should be exported, in the same row of buttons as the `Statistics` button under the `Export Range` heading select:

Custom... -> Day

- **Select `Models` in the lower third of the window.**

Note: You are not required to export the model data set on a regular basis, only when significant changes to the network topology have been made. Exporting, as a daily process, assures that you have a model

name for all data in your statistics data set.

iii. Save the Export Definition Description.

Select:

File -> Save...

In the `Selection` field, type a file name such as `exp_def_desc_1` (SDE automatically appends the `.sde` suffix to the file name), and then press `ENTER` or `RETURN`.

In `exp_def_desc_1.sde`, SDE saves the description of what data to export.

g. Allow SPECTRUM some time to collect data.

If polling and logging have not been enabled until now, SPECTRUM will need about two days to collect a meaningful amount of data. We recommend that you collect two days' worth of data so that you will have a better sense of what typical data are like, what a typical PDB is like, and what typical reports are like as you explore your PDB in later tasks.

h. Start the export process.

After two days, select the following from the current window:

File -> Export Now -> OK

SDE starts the export process, which uses:

- the `exp_def_desc_1.sde` description for what data to export
- the Export Range value of Day for how much data to export
- the directory `SPECTRUM_home/export.output` (by default) for the directory to which SDE will write the data.

In summary, SDE exports all model data to the `model` data set in the `SPECTRUM_home/export.output` directory and exports the previous day's statistics data (logged data) for the selected models to the `statistics` data set in `SPECTRUM_home/export.output`.

- The `model` data set contains model-handle to model-name mappings for the selected model type's models that were present on the previous day. Each observation in the model data set contains the mapping of one model handle (in hex) to its model name (host name).

In a later task you will define a libref named `specgway` that points to the `SPECTRUM_home/export.output` directory. At that point you will refer to the `model` data set as `specgway.model`.

- The `statistics` data set contains the metrics logged during the previous day on the selected model type's models. Each observation in the statistics data set contains a timestamp, a model handle (in hex), and the value at that time of each attribute that was logged for that model handle.

Note: To be exact, the data set that contains the statistics has a name of `stat` not `statistics`. In a later task you will define a libref named `specgway` that points to the `SPECTRUM_home/export.output` directory. At that point, you will refer to the `statistics` data set as `specgway.stat`.

i. Check export progress.

In a separate UNIX window, monitor the log file by using this command:

```
tail -f SPECTRUM_home/export.output/exp_def_desc_1.log
```

where exp_def_desc_1 is the name of the Export Definition Description that you created in this action.

Note: Tail the .log file not the .sde file.

When SPECTRUM displays a message in the UNIX window that indicates the export of the models data and statistics data is complete, press CTRL-C in the UNIX window to interrupt the tail command.

j. Close SDE.

Follow this path:

File -> Close -> OK

If you want to know more about creating and saving an Export Definition Description than was described above, refer to the *SPECTRUM Data Export User's Guide*.

There is also an example of processing SPECTRUM data in the [IT Service Vision Showroom](#).



To set up MeasureWare to monitor your systems, issue this command on each system:

```
mwa start
```

(The MeasureWare command is located in the bin subdirectory of the directory MeasureWare is installed in.)

This will start the MeasureWare SCOPE/UX and other appropriate demons to begin monitoring your system. (This command should probably be part of your system initialization script. See the MeasureWare User's Manual and Installation Manual for details on installing and starting MeasureWare.)

Run the MeasureWare extract utility (from the MeasureWare bin directory) to "export" the metrics to a file. You tell the extract command which metrics you want to export in an extract "report file." An example of this file usually appears in file data/reptall in the directory in which MeasureWare is installed.

although the extraction and processing (by a CSPROCES macro) of the data can theoretically occur at any interval, we recommend daily processing of the data.

For more information on using the extract command, see the man page for extract.

Note that IT Service Vision requires the following additions/updates to the default list of metrics in the example report file.

- **Include DATE and TIME for every data type.**

DATE and TIME are generated automatically in the MeasureWare data record header, but the header value is accurate only to a second. By requesting DATE and TIME explicitly, your DATETIME values in your PDB detail data sets will be more accurate (to hundredths of a second).

- **Do not request HISTOGRAM-type metrics.**

Histogram data is more useful for online quick inquiries into the MeasureWare data rather than for long term, historical analysis with IT Service Vision software. Also, histogram data can produce errors in the extract utility.

- Use **FORMAT BINARY** instead of **FORMAT ASCII**.

IT Service Vision expects the exported data to be in binary format.

- Use **LAYOUT SINGLE** instead of **LAYOUT multiple**.

LAYOUT MULTIPLE data typically contains many placeholder records (ones with zero or negative counts) that are difficult to detect and remove from your PDB and which serve no purpose for long term analysis.

- Do not request **CONFIGURATION** data.

CONFIGURATION data is not supported by IT Service Vision.

When using CSPPROCES to read MeasureWare data, you may get the message "Variable XXXXXXXX is uninitialized" in your initial run with a table. This is an indication that you have a variable that has KEPT=YES in its IT Service Vision dictionary definition, but you have not requested it by means of your MeasureWare report file. The IT Service Vision tables defined for MeasureWare data include metrics for all operating systems supported by MeasureWare and so may include metrics that you are not interested in. Some MeasureWare metrics defined to IT Service Vision have status KEPT=NO. These metrics are typically duplicate information (for example, GBL_DISK_PHYS_IO has KEPT=NO because GBL_DISK_PHYS_IO_RATE is KEPT=YES and is the same information but in the form of a rate). You should review and update your IT Service Vision dictionary table definitions: mark the variables that you are not requesting by means of the report file to have KEPT=NO; mark as KEPT=YES those metrics you want to retain.

For additional information on collecting and analyzing MeasureWare data with IT Service Vision, see the [IT Service Vision Showroom](#).

In file !SASROOT/misc/cpe/cspcsext there is also an example of a script that uses the MeasureWare extract command to export data for reading into an IT Service Vision PDB.



- a. Check that the necessary background processes (daemons) are running on the SNMP management host.

To collect raw data for IT Service Vision from the SNMP management software (in the task after this one), you will use the `snmpCollect` background process. The `snmpCollect` process will use the `ovwdb`, `trapd`, and `ovtopmd` background processes.

- i. Check that these four processes are running.

Issue this command on the management host:

```
ovstatus
```

(If you need more detailed information on this command, type `man ovstatus` at a UNIX prompt or refer to the appropriate HP or IBM documentation listed in the introduction to this document.)

For each background process you will see lines like these. (This is sample output for `snmpCollect`.)

```

object manager name  snmpCollect
behavior             OVs_WELL_BEHAVED
state                RUNNING
PID                 28641
last message        Initialization complete.
exit status          -

```

ii. **Check that the value for state is RUNNING, for each of the four background processes.**

If the state value for one or more of the four processes is not RUNNING, issue this command on the SNMP management host:

```
ovstart
```

If you need more detailed information on this command, type `man ovstart` at a UNIX prompt or refer to the appropriate IBM or HP documentation listed in the [Introduction](#). Re-check the status after issuing the `ovstart` command. If problems with the processes persist, refer to Chapter 4 of *HP OpenView Network Node Manager Administrator's Reference* or to *IBM NetView for AIX Problem Determination*.

b. **Start the SNMP management software graphical user interface.**

```
NetView    nv6000
```

```
OpenView   ovw
```

You can use `ovw` to start the GUI in either case. `nv6000` does some extra housekeeping prior to executing `ovw`.

If you need more detailed information on the `ovw` or `nv6000` command, type `man ovw` or `man nv6000` at the UNIX prompt or refer to the appropriate HP or IBM documentation listed in the [Introduction](#).

Perform all the actions in this task from within the SNMP management application.

c. **Select all the nodes from which you want to collect data.**

i. **Select the IP Internet icon from the root map.**

ii. **Select appropriate nodes.**

On the IP Internet map, while holding down the CTRL key, click Mouse Button 1 (MB1) on every appropriate node (see Prerequisites), and then release the CTRL key. (You may need to select down to a lower layer of the map to reach a map with individual machines.) If you select a node by mistake, you can unselect it by clicking on it again.

Refer to *HP OpenView Windows User's Guide* or *IBM NetView for AIX User's Guide for Beginners* for more information on locating and selecting objects.

You can check your selections by holding down the CTRL key and pressing L. The application displays the items that you selected.

d. **Configure snmpCollect.**

From the main menu of the GUI, take one of the following paths:

NetView Tools -> Data Collection & Thresholds: SNMP
OpenView Options -> Data Collection & Thresholds: SNMP...

The MIB (Management Information Base) data collection window displays.

The top third of this window lists which MIB objects, if any, are already configured for data collection.

Some recommended metrics for your test PDB are listed in [OpenView and NetView Appendix 1: Recommended Metrics](#). To add the selected metrics to your PDB, follow these steps:

i. Select Metrics

Select **Add...**

The MIB object selection window displays. You can make selections by clicking down the object ID (OID) tree in the window until you reach this metric. For example, to select `ifInOctets` follow this path:

mgmt -> mib-2 -> interfaces -> ifTable -> ifEntry -> ifInOctets

Or you can enter the metric's OID directly.

Select **OK**.

The Add Collection window displays.

- Select **Add From Map**.

The nodes that you selected in the previous action are added for this metric.

- Select **Collection Mode** and then **Store, No Thresholds**.
- Type 15m (for 15 minutes) in the `Polling Interval` field.
- Select **Instances** and **All**.
- Select **OK** for your addition to take effect.

The MIB object now displays in the list of configured MIB objects (metrics) in the top third of the screen.

ii. Repeat this sequence for the remaining metrics.

Note: If you find this interactive method too time-consuming, see [OpenView and NetView Appendix 2: Alternate Data-Logging Method](#), for an unofficial but faster method.

e. Begin logging data for the configured objects.

It is important to synchronize the data collection of these objects so that the data are collected very close to the same time for all metrics.

i. Select objects

On the list of configured objects, select all the objects simultaneously by holding down the CTRL key, clicking Mouse Button 1 (MB1) on every object, and then releasing the CTRL key.

ii. Synchronize the times.

Select **Suspend** and then **Resume**.

iii. **Verify that data are being collected.**

You can execute the following command periodically:

```
snmpColDump databases/snmpCollect/ifIndex.n
```

where n is the number of the interface. For instance, if n is 1, this command dumps all the `ifIndex` raw data that were collected until now for interface number 1.

f. **Allow the management software some time to collect data.**

For best results at this stage, allow the management software to collect data for about two days. (At any point after the above action, you can exit the SNMP management application without affecting data logging.) We recommend that you collect two days' worth of data so that you will have a better sense of what typical data are like, what a typical PDB is like, and what typical reports are like as you explore your PDB in later tasks.

At the end of the two days, go on to [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#). You do not need to stop data collection.

There is also an example of processing OpenView and/or NetView data in the [IT Service Vision Showroom](#).



PROBEX data may be logged directly and locally by PROBEX on each system that it is monitoring, or the data may be transmitted to a system running SunNet Manager and logged there. In either case, the format of the logged data is the same (that is, PROBEX logs data in the same format as SunNet Manager). Although in theory PROBEX (or SunNet Manager) will continue to log data until your disk space is totally consumed, we recommend that the log file be moved out of the path of the monitor, processed into your PDB, and then archived to tape fairly frequently (daily).



To start collecting data for IT Service Vision with SunNet Manager, set up SunNet Manager to log the data of interest to disk at some regular interval (usually 15-60 minutes) by using one of the schema files supplied with IT Service Vision. Although in theory SunNet Manager will continue to log data until your disk space is totally consumed, we recommend that the log file be moved out of the path of SunNet Manager, processed into your PDB, and then archived to tape fairly frequently (daily).

If none of the schema files supplied with IT Service Vision will work for your network, you may use any valid SunNet Manager schema file, but you will need to create IT Service Vision dictionary entries from this schema file before reading the data into your PDB. Instructions for defining IT Service Vision tables from schema files appear in [SunNet Manager and Enterprise Manager Appendix 2: Defining Tables from Your Own Schema Files](#).



Set up TRAKKER to log the data of interest to disk at some regular interval (usually 15-60 minutes). Although in theory TRAKKER will continue to log data until your disk space is totally consumed, we recommend that the log file be moved out of the path of TRAKKER, processed into your PDB, and then archived to tape fairly frequently (daily).



To set up MeasureWare to monitor your systems, issue this command on each system:

```
mwa start
```

(The MeasureWare command is located in the bin subdirectory of the directory in which MeasureWare is installed)

This will start MeasureWare's SCOPE/UX and other appropriate demons to begin monitoring your system. (This command should probably be part of your system initialization script. See the MeasureWare *User's Manual* and *Installation Manual* for details on installing and starting MeasureWare.)

Run the MeasureWare extract utility (from the MeasureWare bin directory) to "export" the metrics to a file. You tell the extract command which metrics you want to export in an extract "report file". An example of this file usually appears in file data/reptall in the directory that MeasureWare is installed in.

Though the extraction and processing of the data (by using a CSPPROCES macro) can theoretically occur at any interval, we recommend daily processing of the data.

For more information on using the extract command, see the man page for extract.

Note that IT Service Vision requires the following additions/updates to the default list of metrics in the example report file.

- **Include DATE and TIME for every data type.**

DATE and TIME are generated automatically in the MeasureWare data record header, but the header value is accurate only to a second. By requesting DATE and TIME explicitly, your DATETIME values in your PDB DETAIL data sets will be more accurate (to hundredths of a second).

- **Do not request HISTOGRAM-type metrics.**

Histogram data is more useful for online quick inquiries into the MeasureWare data rather than for long term, historical analysis with IT Service Vision software. Also, histogram data can produce errors in the extract utility.

- **Use FORMAT BINARY instead of FORMAT ASCII.**

IT Service Vision expects the exported data to be in binary format.

- **Use LAYOUT SINGLE instead of LAYOUT multiple.**

LAYOUT MULTIPLE data typically contains many placeholder records (ones with zero or negative counts) that are difficult to detect and remove from your PDB and which serve no purpose for long term analysis.

- **Do not request CONFIGURATION data**

CONFIGURATION data is not supported by IT Service Vision.

When using CSPROCES to read MeasureWare data, you may get the message "Variable XXXXXXXX is uninitialized" in your initial run with a table. This is an indication that you have a variable that has KEPT=YES in its IT Service Vision dictionary definition, but you have not requested it by means of your MeasureWare report file. The IT Service Vision tables defined for MeasureWare data include metrics for all operating systems supported by MeasureWare and so may include metrics that you are not interested in. Some MeasureWare metrics defined to IT Service Vision have status KEPT=NO. These metrics are typically duplicate information (for example, GBL_DISK_PHYS_IO has KEPT=NO because GBL_DISK_PHYS_IO_RATE is KEPT=YES and is the same information but in the form of a rate). You should review and update your IT Service Vision dictionary table definitions: mark the variables you are not requesting by means of the report file to have KEPT=NO; and to mark as KEPT=YES those metrics that you want to retain.

For additional information on collecting and analyzing MeasureWare data with IT Service Vision, see the [IT Service Vision Showroom](#).

In file !SASROOT\misc\cpe\cspcsext there is also an example of a script that uses the MeasureWare extract command to export data for reading into an IT Service Vision PDB.



a. Check that the necessary services are running on the SNMP management host.

To collect raw data for IT Service Vision from the SNMP management software (in the task after this one), you will use the `snmpCollect` service. The `snmpCollect` service will use the `ovwdb`, `trapd`, and `ovtopmd` services.

i. Check that these four services are running.

To check the NNM services status issue the following command from a DOS command window:

```
ovstatus
```

For each service you will see lines like these. (This is sample output for `snmpCollect`.)

```
object manager name  snmpCollect
behavior             OVs_WELL_BEHAVED
state                RUNNING
PID                  28641
last message         Initialization complete.
exit status          -
```

- ii. **Check that the value for `state` is `RUNNING`, for each of the four background processes.**

If the `state` value for one or more of the four services is not `RUNNING`, issue the following command from a DOS command window:

```
ovstart
```

- b. **Start the SNMP management software graphical user interface.**

Start->Programs->HP OpenView->Network Node Manager

Perform all the actions in this task from within the SNMP management application.

- c. **Select all the nodes from which you want to collect data.**

- i. **Select the `IP Internet` icon from the root map.**

- ii. **Select appropriate nodes.**

On the IP Internet map, while holding down the CTRL key, click Mouse Button 1 (MB1) on every appropriate node (see Prerequisites), and then release the CTRL key. (You may need to select down to a lower layer of the map to reach a map with individual machines.) If you select a node by mistake, you can unselect it by clicking on it again.

Refer to *HP OpenView Using Network Node Manager* for more information on locating and selecting objects.

You can check your selections by holding down the CTRL key and pressing L. The application displays the items that you selected.

- d. **Configure `snmpCollect`.**

From the main menu of the GUI, take one of the following paths:

Options -> Data Collection & Thresholds: SNMP...

The MIB (Management Information Base) data collection window displays.

The top third of this window lists which MIB objects, if any, are already configured for data collection.

Some recommended metrics for your test PDB are listed in [OpenView and NetView Appendix 1: Recommended Metrics](#). To add the selected metrics to your PDB, follow these steps:

- i. **Select Metrics**

Select **Add...**

The MIB object selection window displays. You can make selections by clicking down the object ID (OID) tree in the window until you reach this metric. For example, to select `ifInOctets` follow this path:

mgmt -> mib-2 -> interfaces -> ifTable -> ifEntry -> ifInOctets

Or you can enter the metric's OID directly.

Select **OK**.

The Add Collection window displays.

- Select **Add From Map**.

The nodes that you selected in the previous action are added for this metric

- Select **Collection Mode** and then **Store, No Thresholds**.
- Type 15m (for 15 minutes) in the `Polling Interval` field.
- Select **Instances** and **All**.
- Select **OK** for your addition to take effect.

The MIB object now displays in the list of configured MIB objects (metrics) in the top third of the screen.

ii. **Repeat this sequence for the remaining metrics.**

Note: If you find this interactive method too time-consuming, see [OpenView and NetView Appendix 2: Alternate Data-Logging Method](#), for an unofficial but faster method.

e. **Begin logging data for the configured objects.**

It is important to synchronize the data collection of these objects so that the data are collected very close to the same time for all metrics.

i. **Select objects**

On the list of configured objects, select all the objects simultaneously by holding down the CTRL key, clicking Mouse Button 1 (MB1) on every object, and then releasing the CTRL key.

ii. **Synchronize the times.**

Select **Suspend** and then **Resume**.

iii. **Verify that data are being collected.**

You can execute the following command periodically:

```
snmpColdDump databases\snmpCollect\ifIndex.n
```

where *n* is the number of the interface. For instance, if *n* is 1, this command dumps all the `ifIndex` raw data that were collected until now for interface number 1.

f. **Allow the management software some time to collect data.**

For best results at this stage, allow the management software to collect data for about two days. (At any point after the above action, you can exit the SNMP management application without affecting data logging.) We recommend that you collect two days' worth of data so that you will have a better sense of what typical data are like, what a typical PDB is like, and what typical reports are like as you explore your PDB in later tasks.

At the end of the two days, go on to [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#). You do not need to stop data collection.

For other collectors, see the applicable example in the [IT Service Vision Showroom](#).

Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data

Create a test PDB and add one or more tables to it or create tables in it. Process your data into the table(s), reduce the data, generate reports on the data, and view the reports.

Prerequisites

1. **You have logged data for approximately six hours to one day, depending on the amount of data that you want to collect.**
2. **You have determined which day is considered the start of the week at your site.**

Since some data in IT Service Vision performance data bases is summarized and reported by week, you will need to determine what weekday should be start-of-week for your data. At some sites, a week is defined to run from Sunday through Saturday; other sites define a week to run from Monday through Sunday; and so on. The default start-of-week is Sunday.

3. You may want to check with other performance areas at your site to check that all areas are using the same day as the start of the week so that all the reports are consistent.

Note: When you are working with the test PDB, as you are in this task, it is convenient to do the actions interactively. Thus, this task is described in terms of the IT Service Vision GUI.

When you are working with the production PDB, as you will be in Section 2, it is convenient to schedule the steps as a job in background (using a task scheduler, such as cron on Unix or the System Agent on Windows). Thus, the corresponding task in Section 2 will be described in terms of a background job and scheduled as a background job.

In order to have a background job to work with in Section 2, this task will save the background form of the interactive job. After you save it, you can ignore it until Section 2 refers to it.

Actions

1. Create your test PDB.

a. Allocate

i. Invoke the Create PDB wizard.

From the main window of the IT Service Vision server interface, select

Administration -> Create PDB - Wizard

ii. Specify the physical name of your test PDB.

Type the physical name of your test PDB, without any reference to the server on which it will reside. We suggest storing your test PDB in:

UNIX /tmp/pdb-*name*

WNT C:\temp\pdb-*name*

Where *name* is a unique name of your choosing (in the examples in this document, we will use a short

name derived from the collector name).

In the Wizard, select **Next** to Continue.

b. Add or create table and variable definitions.

i. Use the IT Service Vision user interface to add the tables and variables.

Select **Select Tables**. IT Service Vision displays the Select Tables window.

ii. Select the table(s) based on your collector.

From the collector list, select your collector.

In the table list select your table(s).

- There are special instructions for selecting tables for the following data sources:



Scroll down to the HN2IFT table and the HN2NIX table and select (click on) these two tables.

Select **OK**.



If you are using the HubCSIMRXi model type, scroll down to the CSIMRX table and select (click on) the table name. If you are using a different model type, scroll down to the appropriate table for that model type and select the table name.

Select **OK**.



Select the appropriate tables for your SunNet Manager data based on the schema file used to collect the data. If you are using schema files supplied by IT Service Vision, you can select your tables by matching the description displayed in the **Select Tables** list. See [SunNet Manager and Enterprise Manager Appendix 1: Mapping SunNet Manager Metrics to IT Service Vision Tables and Variables](#) for instructions for this matching.

If you decided to use one of your own schema files instead of an IT Service Vision supplied schema file, the table names do not appear in the **Select Tables** list. You need to create your own IT Service Vision table definitions from your schema file. Instructions for defining IT Service

Vision tables from schema files appear in [SunNet Manager and Enterprise Manager Appendix 2: Defining Tables from Your Own Schema Files](#). The table definition process adds the tables directly to your PDB (so you do not select the tables from the **Select Tables** list).

- For other data sources, select the tables in the **Select Tables** list that correspond to the data in which you are interested by matching the table description displayed to your collector vendor's description of the data. In the [IT Service Vision Showroom](#) there are examples of which tables to select for many collectors.

Click on **OK** to complete selection of the tables.

To see a list the of selected tables, choose `List Selected Tables`. Exit to the previous screen by selecting **Close**.

In the wizard, select **Next** to continue.

c. Select the start of week.

Specify the start of the week for your test PDB. The default is Sunday.

Select **Next** to continue.

d. Specify if you want the new PDB to become the active PDB after it is created.

For this setup example, select **yes** and then select **Next** to continue.

e. Verify PDB Options.

After you have selected all your PDB options, the wizard will now display a summary list to confirm your choices. Select **Finish** to continue.

2. Process and reduce your data.

a. Start the process data wizard.

From the main window, select **Process Data - Wizard**.

Follow the directions in the process wizard. You will need to know the following information:

- information on your collector
- which tables to process
- the location of the raw data file
- any optional or advanced commands for your collector that you want to specify at process time.

At the end of the process data wizard, you have an opportunity to save the source code for this invocation of the process macro. Save the source code; you will use it in Section 2. It will be necessary for the batch processing examples in this document.

Additionally, after processing, you have the option of continuing with the reduction wizard.

Check the SAS LOG window for any messages, warnings, or errors, in the process step.

b. Start the reduce data wizard.

The reduce data wizard can be started in two ways:

- from the main windows, select **Reduce Data - Wizard**
- at the end of the process data wizard, continue with the reduce data wizard.

Follow the directions in the reduce data wizard. At the end of the reduce data wizard, you have an opportunity to save the source code for this invocation of the reduce macro. Save the source code; you will use it in Section 2. It will be necessary for the batch processing examples in this document.

Check the SAS LOG window for any messages, warnings, or errors, in the reduce step.

3. Generate one or more reports on the data in your test PDB.

List the supplied report definitions that are based on the tables in your test PDB. Follow this path from the IT Service Vision main window:

Reports -> Manage Report Definitions

IT Service Vision displays a list of supplied report definitions that are based on the tables in your PDB.

The name of each report definition follows a naming convention. A simplified description of the naming convention can be found in [Shared Appendix 7: Supplied Reports Naming Convention](#). Each report definition is intended for a specific audience. For details about the associated audiences, refer to the Macro Reference documentation for IT Service Vision.

Note: If there are no supplied reports for your collector/data source, make at least one report definition. For information on creating report definitions, in the Manage Report Definitions window, select **Online Help -> Help on this window**. For convenience use the Plots report style. You may find this table helpful; it describes the mechanics of creating a simple report definition in the Manage Report Definitions window.

Select
Select Table's right arrow -> Levels Detail -> level.table DETAIL. TABLENAME -> OK
Select Variable's right arrow -> Variable Types Analysis -> <i>Variable Name</i> -> Add
Attributes -> <i>Variable Name</i> -> type Variable Label <i>Variable Label</i> -> OK
Variable Types Statistics -> MEAN -> Add
Variable Types By -> MACHINE -> Add
Variable Types X Variable -> DATETIME -> Add
Attributes -> DATETIME -> type Variable Label Date -> OK -> OK
Summary Time Period's right arrow -> AS IS -> OK
click on image under Current Report Style -> Report Style Plots -> Report Sub-Style Line -> OK

To save this report definition your SASUSER.ITSVRPT folder, follow this path from the Manage Report Definitions window: **File -> Save Report Definition**.

- **Read about the report and view an example report for each report definition.**

Note: This will not apply if you just created a new report definition above. Continue at the bullet "Select and run...".

i. **Read the purpose of the report.**

From the list of supplied reports, follow this path:

select the report definition(s) -> **Locals** -> **Show report explanation**

IT Service Vision displays the explanation in a separate window. Note whether the report requires some

preparation (data manipulation) prior to running it. (This is listed after the heading **PreReqs** in the Explanation.) In a later step you will run the reports that do not require setup. Close the window.

ii. **View sample output from the report.**

Follow this path for the same report definition:

Locals -> Show sample output

Close the window.

o **Select and run each report definition that does not require setup.**

Follow this path:

select the report definition -> Run



Note: Reports HRBTC01 and HRBTC02 contain information on IP gateway data. If your PDB does not have gateway data, you can ignore these reports definitions. If you try to run these report definitions and there is no gateway data in your PDB, a window pops up and displays a message that no observations apply to this report.

o **Save the report results (graph report or text report) of running the report definition.**

At the command prompt, type BUILD. At the SAS BUILD window, select WORK under the Libraries scroll box. Then select GSEG in the Catalogs scroll box. Your new report definition titles are displayed in the Entries scroll box.

To save your report definitions:

select all reports to be saved -> File -> Copy...

In the To selection of the Copy dialog box, type the name of the SAS catalog where you want to save the report (use SASUSER.GSEG for now). Then select **File** and **End**.

To return to the Manage Report Definitions window, select **File** and **End**.

Note: By default, when you save a graphics or text report, IT Service Vision stores them in your SASUSER library. In the next section you will learn how to save them elsewhere.

4. **Display your reports.**

You may also want to show your reports to your manager as confirmation that you successfully collected, processed, reduced, and reported on the data from your site.

The reports have been saved in a SAS catalog. To get access to the SAS catalogs and your reports, type BUILD at the command prompt. Select **SASUSER** under the Libraries scroll box, and then **GSEG** under the Catalogs scroll box.

Then follow this path:

select report to view or edit -> **File** -> **Open** -> **Browse** or **Edit**

Section 2, Making the Transition to Production

Section 1 got you started with data collection, a test PDB, and some simple report definitions. This section describes how to finish customizing the data collection, PDB, and report definitions to the specific requirements at your site. It also describes how to put the customized versions into production.

Section 2, Task 1: Customize and Verify Your Test PDB

This task walks you through planning which variables to keep in your production PDB. It also walks you through implementing decisions in the test PDB, checking the test PDB, and adjusting the test PDB until you are satisfied with your choices. In the task after this one, you will make a production PDB from the customized test PDB.

Prerequisites

1. You have a printed copy of the table definition(s).

It is convenient to have a copy of all the table definition(s) that are in your PDB in front of you as you step through the actions in this task.

To print from the IT Service Vision interface:

a. Clear the SAS log.

Unless you want to save a complete log of your SAS session, you may want to clear the log before you display the definitions on it. If so, make the SAS log window the active window and then select **Edit** and **Clear** text.

b. Selecting the table(s).

- Follow this path from the IT Service Vision main window:

Administration -> Manage Tables

- Select the tables that you defined when you created your PDB.
- Click the right mouse button and then select **Print Table Definition**.

In the log window, IT Service Vision displays a copy of the table definition (and of the table's variable definitions).

If you want to find specific information in the LOG window follow this path:

Edit -> Find...

IT Service Vision displays the Find...: LOG window. Type a search string in the Find field. Select the other search parameters, and then select **Find**. IT Service Vision scrolls the log to the specified string. To find the next instance, select **Find** again.

c. Copy the table definition(s) to an external file.

Follow this path:

File -> Save as -> Write to File...

Type the full pathname of the file in the Enter directory, filename, or filter field, and select **OK**.

d. Use your local print facility to print the file.

To print in background/batch:

i. Create a file containing these statements:

```
* Printing a table definition in the SAS log;
%CPSTART( pdb=pdbname, mode=batch );
%PCAT; cards4;
print table name=_ALL_;
/* The four semicolons must be in column one */
;;;;
%PCAT( cat=work.temp.prtab.source );
%CPDDUTL( entrynam=work.temp.prtab.source, list=yes );
```

where *pdbname* is the name of your PDB. For more information on the PRINT TABLE control statement, see the %CPDDUTL control statements in the Macro Reference documentation for IT Service Vision.

ii. Start SAS from the UNIX shell command prompt as follows:

```
sas filename -terminal -batch -dmsbatch -fsdevice
ascii.vt100
```

where filename is the name of the above file. Or, type the contents of the above file (if the IT Service Vision server software is already running, omit the invocation of the %CPSTART macro) starting at the first line in the body of your PROGRAM EDITOR window. Submit the above file for SAS to run.

The control statements display in the SAS log, followed by the output that they generate. To print the log, see the instructions earlier in this prerequisite.

2. In the tables that you chose, you have determined which variables you want to use and you have reviewed the default settings of the variables and the variables' statistics.

On the hardcopy versions of the table definitions, mark in what you want to change, if anything, based on the actions below. Later in this task you will implement these changes.

a. Review the BY variables list and CLASS variables lists for each of the tables that you selected.

The kept status of the variables in the BY variables list and CLASS variables list must be Yes.

The BY variables list applies to the detail level of the table. The purpose of the BY variables list is to sort the observations in the detail level to match the sort order of the observations incoming during the processing step and to detect observations that have duplicate values in the BY variables.

Under normal circumstances, you do not need to modify the BY variables list, but if you need to modify the list you can. For example, if you add a variable in an exit, you may (or may not) want to add that variable to the BY variables list.

The variables on the list must have a Kept status of Yes. DATETIME must be on the BY variables list. To the right of DATETIME on the list, use only HOUR and/or SHIFT. To the left of DATETIME on the list, you can use any variables.

CLASS variables lists, displayed in the bottom of the Edit Table window, are used to group data so that the same values of all CLASS variables are together for the reduction step. The lists also assure that data are in the proper sort order for reporting on the reduction levels. By default, the CLASS variable lists contain the same variables in the same order as the BY variables list, but the CLASS variables list in your tables may be different. The CLASS variables list for a reduction level can be different from the BY variables list and can be different from all other CLASS variables lists, but it must contain the DATETIME variable.

If you want character string metrics like MACHINE and JOB to be carried into reduction levels for reporting, add them to the CLASS variables lists immediately to the left of the DATETIME variable. (This presumes that you changed the Kept status of the string variable to Yes earlier. However, even if their Kept status is Yes, string variables that are not on the CLASS variables list are dropped from the non-detail levels because there are no statistics for string variables.)

Each of the BY and CLASS variables must have a Kept status of Yes.

Note: These lists apply to a table definition and are used to process and reduce data.

This BY variables list is not related to the BY variables that you can select for a report definition.

More about frequently used variables:

- MACHINE is the host name of the node on which the metrics were created. In the process step, the value of MACHINE is obtained from the value of IPADRES.
- DATETIME is the timestamp on the observation.
- IPADRES is the TCP/IP address of the machine (node) that created the metric values in an observation. To group data, IT Service Vision uses MACHINE name (host name) rather than IPADRES because IP addresses generally change more frequently than system names. (IP addresses can change for network upgrades, office moves, and so on.) Generally, the mapping of IPADRES to a MACHINE name happens automatically (via snmpColDump) at the process step by using the management host's IP address table or name server. If data for an IP address is encountered in the input data for a machine that has not been defined yet to the SNMP management host, MACHINE will be assigned the same value as IPADRES's value.
- INSTANS is the interface identifier. For example, a router may have two ports, each with a unique instance identifier such as 0 or 1.
- If you want reports based on IPADRES instead of MACHINE, you can make them in a task later in this section. That task walks you through how to make custom report definitions by modifying supplied report definitions and saving them.

Note: IPADRES, and INSTANS apply to HP OpenView and NetView for AIX only.

- HOUR is the hour of the observation.

If the Kept status of HOUR is Yes, when an observation is processed into the PDB then IT Service Vision calculates the value of HOUR. The value of hour is based on the hour in the value of DATETIME.

If HOUR is in the BY variables list, you must keep data for it. If HOUR is not in the BY variables list, you can set the Kept status of HOUR to No; however, that change is not recommended.

- SHIFT is the work shift code for the observation.

If the Kept status of SHIFT is Yes, when an observation is processed into the PDB then IT Service Vision calculates the value of SHIFT (unless the observation has a non-blank value for SHIFT). First, IT Service Vision checks to see if the date in DATETIME is on the list of holidays at your site. If so, the value of SHIFT is set to the work shift code for holidays. If not, the value of SHIFT is based on the day-of-week and hour of the value that is in DATETIME and on the grid that contains work shift code by Day of Week and Hour of Day. For more details on the work shift grid, see [Section 1, Task 1: Start IT Service Vision Server Software](#).

If SHIFT is in the BY variables list, you must keep data for it. If SHIFT is not in the BY variables list, you can set the Kept status of SHIFT to No; however, that change is not recommended.

b. Review the ID variables lists for each of the tables that you selected.

ID variables are non-changing or slowly changing variables whose values you want to have available for reporting at one or more reduction levels.

c. Review the Index variables lists for each of the tables that you selected.

Index variables are variables whose values you want to index at one or more level. The purpose of an index is to speed up the generation of reports when you run a report definition that has one or more of the indexed variables.

d. **Review the other variables in your table to check whether their Kept status and statistics to be calculated are appropriate for your site.**

Within your table(s), the variables that are generally most useful for performance analysis are the ones whose default values for the Kept status are Yes. These are the variables that you logged in the first section of this document.



If in the SAS log, you received messages similar to this:

```
NOTE: CSIIPDI not initialized
```

while running %CSPROCES on the exported data, then some of the variables defined (in the IT Service Vision table definition) for this device are not being logged. To enable logging of these attributes, see [Cabletron Spectrum Appendix 5: Editing Model Types for SPECTRUM](#).

Note: If you decide on a Kept status of Yes, you can decide later to change it to No, in which case the values are not processed into the detail level from that time forward. Similarly, if you decide on a Kept status of No, you can decide later to change it to Yes, in which case values are processed into the detail level from that time forward.

e. **Check the age limit of each table.**

i. **Check the Age Limit at detail level.**

By default, IT Service Vision keeps ten days of data at the detail level, but the age limit in your tables may be different. If you want your reports on detail level to be able to cover a different number of days (for example, 14 days), overwrite the current value with the new value (in this case, 14).

Note: If you do not want to generate reports on detail-level data in this table but you do want to generate reports on other levels, you can set the value to 0.

Note: The process step deletes *existing* data in the detail level that are beyond the age limit for the detail level, but the process step keeps *incoming* data (and adds it to existing data in the detail level) regardless of the dates in the incoming data. When the age limit at detail level is 0, the *process step* behaves just as it always does: it deletes existing detail level data whose age is greater than the age limit (thus, in this case, it deletes all of the existing detail level data), and it keeps the incoming data in the detail level (thus, in this case, what were the incoming data are now all the existing data).

The reduce step ages out data in non-detail levels that are beyond the age limits for those levels and uses the existing data in detail level to update the non-detail levels. If and only if the age limit at detail level is 0, the *reduce step* also deletes the existing data (which, if age limit is 0, are the data that were incoming data to the process step) in the detail level.

If the age limit is 0, the existing data in detail level would be cleared anyway (at the time of the next *process step*) because the data by now are existing data and over the age limit for detail level. The purpose of this exception is to remove the existing data in detail level many hours sooner. This is quite useful when there is a tremendous volume of detail-level data and the data can be 'thrown away' after the day, week, month, and/or year levels have been updated. Performing this frees extra disk space between

production jobs when the data will not be used again.

There is one *caution*: if you have a failure (for instance, a power outage) during the reduce step and any age limits at detail level are zero and any age limits at non-detail levels are non-zero, make sure that you re-submit the reduce step before you go on; otherwise, the *next* day's production job will remove the detail data before the data have an opportunity to be summarized in the non-detail levels.

ii. Check the Age Limits at reduction levels.

By default, the PDB keeps reduction-level data as follows

Day	45	days
Week	15	weeks
Month	18	months
Year	5	years

but the age limits in your tables may be different.

If you want your reports on these levels to be able to cover a different number of days, weeks, months, or years, or if you want to keep fewer data in order to save disk space and shorten processing time, overwrite the current value with a new value.

f. Consider whether to keep data for string variables whose default value is kept status in No.

The following variables/metrics fall in this category: IFSPEC (ifSpecific), SYSCONT (sysContact), SYSLOC (sysLocation), SYSNAME (sysName), and SYSOBID (sysObjectID).

The process step takes longer when string variables are kept in the detail level of the PDB. HPOV/NVAIX does not automatically collect the values of the string variables so their Kept status is set to No by default. If you change the Kept status to Yes, an additional snmp get request per variable is required to gather this data during the process step.

Gathering this data can slow down the process step, especially if there are network problems that delay the snmp get requests. However, the information in these fields is often useful in reports; for example, an interfaces report on a router is more meaningful if the interface is identified as "Ethernet/0" rather than just "4".

If you are thinking of keeping values for string variables, in the task after this one you can time processing with the Kept status set to various combinations of Yes and No to check how much extra time is required to gather the data. Based on that, you can decide whether the time to gather the data is acceptable to you.

3. Your test PDB is the active PDB, and you have write access to it.

4. If you want to monitor line utilization for full-duplex lines, you know the host name and instance number of each full-duplex interface, if any, on your network.

For example, you would know something like this: the number 2 interface on dimes.unx.sas.com and the number 4 interface on swimmer.unx.sas.com are full-duplex, and these are the only full duplex interfaces.

Actions

1. If you plan to customize the PDB, delete data from the PDB.

You can customize with data in the PDB, but deleting the data makes the customization and testing much faster and easier. Deleting the data also results in a consistent set of data at the end of this task.

a. Display the list of tables.

Follow this path from the IT Service Vision main window:

Administration -> Manage Tables

IT Service Vision displays a list of the tables in the active PDB.

b. Delete the data.

Follow this path:

Manage Tables-> select a table -> right button to bring up popup menu -> select 'Delete Table' -> Delete data only

c. Return to main window.

File -> End

2. As necessary, make changes to your PDB's work shift schedule, holiday list, and start of week.

For the work shift schedule and holiday list, see [Section 1, Task 1: Start the IT Service Vision Server Software](#), except make two passes: when working on the work shift schedule, select **Active PDB**, make changes, and **OK** out (or select **Site Default**, make changes, **OK** out, return, select **Active PDB**, select **Copy from SITELIB**, and **OK** out); when working on the holiday list, select **Site Default**, make changes, and **OK** out.

For the start of week, see Action 1 of [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#).

3. As necessary, edit the table and variable definitions for each table in your test PDB.

a. Display the table definitions or table properties window.

Follow this path:

Administration -> Manage Tables -> click right mouse button on the table -> Properties

b. Change the Table Kept status of the table, if necessary.

On the General tab, Table Kept is in the upper right quadrant.

Check that Table Kept status is set to Yes if you want to keep data in the table.

c. Change the Age Limits of the levels in the table, if necessary.

On the Table Status tab, see the intersection of the Level row and Age Limit column, for each of the levels (Detail through Year).

To change a value, overwrite it.

d. List the variables in the table.

Note: The term metric means the same as the term variable. Typically, the term metric is used for a field on the records in the data log, and the term variable is used for a field on the observations in a table in the PDB.

On the General tab, select **Variables**, which is on the right side of the window.

IT Service Vision displays a list of the variables in this table.



If you decide in the task before this one, to change the SPECTRUM attributes for the model/table in your PDB, now is the time to add variables that correspond to these metrics. See [Cabletron Spectrum Appendix 6: Mapping SPECTRUM Data Types to an IT Service Vision PDB](#) in this document for information on adding new variables to a table.

To add new variable(s) in the View Variables window, follow this path:

select variables -> close

then follow the information under [help] on each tab.

To return to the list of variables, select **OK**.

e. Adjust the Kept status of the variables, if necessary.

Look through the decisions that you made in the prerequisites to this task for the variables for which you now *want* to keep data. For that set of variables, find the ones whose Kept status is currently No. To change the Kept status for all these at the same time, follow this path from the list of variables:

select list of variables to change from No to Yes -> right mouse click -> Properties -> Modify the kept in PDB status to Yes

Data will be kept at detail level for these variables.

Look through your list for the variables for which you *no longer want* to collect data. For that set of variables, find the ones whose Kept status is currently Yes. To change the Kept status for all these at the same time, follow this path from the list of variables:

select all variables to change from Yes to No -> right mouse click -> Properties -> Modify the Kept in PDB status to No

Data will not be read and thus not kept at detail level for these variables. As a consequence, data will also not be available at reduction levels for these variables.

f. Adjust the calculation status of the statistics on the variables whose Kept status is Yes, if you require values different from the default settings.

One of the attributes of a variable is its interpretation type, such as INT, C2RATE, STRING, TIME, and so on. By default, IT Service Vision calculates the same statistics for all variables of the same interpretation type. For a table of the default statistics that correspond to each interpretation type, see [Shared Appendix 6: Characteristics of Variables](#).

select the variable -> click right mouse button -> Properties -> Statistics

If you want to change which statistics are set for a variable, IT Service Vision displays a grid of statistic vs. level. Highlighted statistics are calculated at the indicated level.

For all the variables, if you want to see which statistics are selected it is often easier to look at the complete table definition (see the Prerequisites in this task). Note that in a variable definition, the statistics are listed by level.

Select or de-select statistics to correspond to your choices. (Only the selected statistics will be calculated.) Then select **OK**.

When you finish editing the statistics for the variables whose Kept status is Yes, return to the table definition by selecting **OK**.

g. **Modify the BY and Class variables lists, ID variables lists, and index variables lists, if you require values different from the default settings.**

The BY variables list specifies the sort order and grouping of data at the detail level. Each Class variables list specifies the sort order and grouping of the data used to calculate the statistics at a reduction level (day, week, month, or year).

Each ID variables list specifies the non-changing (or slowly changing) variables whose values you want to have available at one or more reduction levels.

Each index variables list specifies the variables whose values you want to index, in order to speed up report generation when you run a report definition that uses one or more of those variables.

Select the Structure Variables tab on the Table Properties window.

i. To change the BY and/or Class variables lists, select **BY/Class**.

Now select the level button for a level whose list you want to change. (Select Detail for the BY list; select one of the reduction levels for that level's CLASS list.) IT Service Vision displays a window that has two lists. The upper list is the current BY or CLASS variables, in the numbered order. The lower list is the remaining variables in the table.

To remove a variable from the upper list, select the variable and click on the down arrow. The variable will be moved to the lower list. To add a variable to the upper list, select the variable and click on the up arrow. The variable will be moved to the upper window.

When the numbered list contains the correct variables in the correct order, select **OK** to return to the Table Properties window.

ii. To change one or more ID variables lists, select **ID**.

Now select the level button for a level whose list you want to change. IT Service Vision displays a window that has two lists. The upper list is the current ID variables. (Order has no meaning for ID variables.) The lower list is the remaining variables in the table.

To remove a variable from the upper list, select the variable and click on the down arrow. The variable will be moved to the lower list. To add a variable to the upper list, select the variable and click on the up arrow. The variable will be moved to the upper window.

When the lists contain the correct variables, select **OK** to return to the Table Properties window.

iii. To change one or more Index variables lists, select **Index**.

Now select the level button for a level whose list you want to change. IT Service Vision displays a window that has two lists. The upper list is the current Index variables. (Order has no meaning for Index variables.) The lower list is the remaining variables in the table.

To remove a variable from the upper list, select the variable and click on the down arrow. The variable will be moved to the lower list. To add a variable to the upper list, select the variable and click on the up arrow. The variable will be moved to the upper window.

When the lists contain the correct variables, select **OK** to return to the Table Properties window.

h. **Save your customizations of the tables and variables.**

After you finish customizing a table and its variables, on the table definition/properties window select **OK**.

It can take a long time for the data dictionary to be updated and the view or views to be recreated, perhaps several minutes. If you have time to do this now, select **OK**; otherwise, select **Cancel**. (The next action describes what to do later if you select **Cancel** now.) Then IT Service Vision returns you to the list of tables in your PDB.

After the last table, return to the main menu by selecting **File** and then **End**.

4. Build views.

Note: If you canceled out of view update, you will need to update your PDB views in background/batch if you want to do reporting before the next time that you run the process step. (The process step updates views, as necessary.)

a. Create a file containing these statements.

```
* Building views;
%CPSTART( pdb=your_test_pdb, mode=batch, access=write );
%CPCAT; cards4;
build views name=_all_ ;
/* The four semicolons must start in column one */
;;;
%CPCAT( cat=work.temp.bldviews.source );
%CPDDUTL( entrynam=work.temp.bldviews.source, list=yes );
```

where *your_test_pdb* is the name of your PDB, %CPCAT will load a catalog entry with a control statement and %CPDDUTL will apply that control statement to the PDB.

For more information on the BUILD VIEWS control statement, see the Reference Documentation for IT Service Vision.

b. Start SAS from your operating system prompt as follows.

UNIX `sas filename -terminal -batch -dmsbatch -fsdevice ascii.vt100`

WNT `start /w c:\m612_win\sas -sysin filename -dmsbatch -icon`

where *filename* is the name of the file that contains the above statements.

Or, submit the statements (if the IT Service Vision server software is already running), in the PROGRAM EDITOR window. Type the set of statements and select **Locals** and **Submit**.

5. Log and collect another day's data.

As necessary, modify the logging and collection software to provide the collected metrics.

For details, see [Section 1, Task 2: Start Your Data Collection Software](#). Remember to keep the polling interval short enough that at least two full polling intervals occur between process runs.

Wait approximately six hours to one day for additional data to be logged and collected if you have modified the logging and collection software.



Note: This Task describes adding a metric by clicking down the OID tree. It is somewhat faster to type the metric's OID in the MIB Object Id field and press ENTER or RETURN. To obtain the OIDs for the metrics that you want to change, refer to your copy of the complete table definition (see Prerequisites for this task).

6. Process and reduce a day's data.

Approximately six hours to one day after you changed the configuration of your PDB and collector (to give your collector the time to collect data based on the new configuration), process and reduce data again. For details, see [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on the Data](#).

Note: If you are considering keeping some additional string variables, time the process step. Then reverse the Kept settings on these variables and time the process step six hours to a day later. You should be able to decide from this whether you want to keep any of these variables.

The delay caused by getting values for these variables depends on your data volume, the number of nodes reporting data, and your network's responsiveness. The only way to evaluate the impact of keeping these variables is to time them in *your* process step.

7. Check the results.

a. Examine the data.

From the main window, follow this path:

PDB Administration -> Examine PDB Data -> *select your tables* -> right mouse click -> View Data

Use the scroll bars to move around in the data.



Depending on the vendor of a network device, some metrics are set by hand at configuration time. To assure that they were typed correctly, check the configuration of your network devices to assure that the following variables have appropriate values:

- IFDESC, whose values are used to identify interfaces (if this string variable has Kept status of Yes in your PDB)
- IFSPEED, whose values are crucial in calculation utilization (if this string variable has Kept Status of Yes in your PDB)
- any other variables that were entered manually when the network devices were configured.

In table HN2IFT, the calculation of `ifUtilization` (IFUTIL) in IT Service Vision uses the value of `ifSpeed` (IFSPEED). Frequently, `ifSpeed` is established for each interface of each network device manually by the technician configuring the device. There is no automatic validity check; the value could be incorrect. Using other configuration information that you have, you need to assure yourself that the values being reported for `ifSpeed` are indeed correct.

Another consideration of using `ifSpeed` and `ifUtilization` is determining whether an interface is half-duplex or full-duplex. In the calculation of `ifUtilization` that is supplied with IT Service Vision, all interfaces are assumed to be half-duplex. Interface utilization is thus calculated using the sum of input and output octets

$$\text{ifutil} = (\text{ifiocts} + \text{ifoocts}) * 8 / \text{ifspeed}$$

If you use concentrators on your lines or if the interface is full-duplex, utilization may sometimes be reported as exceeding 100%. For a line with concentrators, this is the only meaningful indicator of how much of the capacity of the line is being consumed. However, the utilization of full-duplex interfaces is more meaningfully reported as the maximum of input and output octets

$$\text{ifutil} = \max(\text{ifiocts}, \text{ifoocts}) * 8 / \text{ifspeed}$$

To identify to IT Service Vision which of your interfaces are full-duplex, replace the supplied SAS informat definition (which presumes all interfaces are half-duplex) with your own definition that identifies which interfaces at your site are full-duplex. Follow these steps:

- a. Check that you started the IT Service Vision server with the `SITEACC=` parameter set to write.
- b. In the PROGRAM EDITOR window, type this program starting at the first line in the body of the window

```
proc format library=sitelib.cpfmts;
invalue CSFDX
'hostname instance' = 1
'hostname instance' = 1
.
.
.
other = 0;
run;
```

where *hostname* and *instance* identify a full-duplex interface and a single blank separates them. The reference to `sitelib` indicates that this informat is being stored in your `SITELIB` library and will be accessible to everyone at your site.

For example, suppose your IT Service Vision site library is `/usr/cpe/sitelib`, the number 2 interface on `dimes.unx.sas.com` and the number 4 interface on `swimmer.unx.sas.com` are full-duplex, and all other machines and interfaces are not full-duplex. Your informat definition program would look like this:

```
proc format library=sitelib.cpfmts;
invalue CSFDX
'dimes.unx.sas.com 2' = 1
'swimmer.unx.sas.com 4' = 1
other = 0 ;
run;
```

- c. Submit the above commands.
- d. To check the format, type this program in your PROGRAM EDITOR window, starting at the first line in the body of the window:

```
proc fsvview data=detail.hn2ift;
where input(trim(machine)||"
"||left(put(instans,7.)),CSFDX.);
run;
```

- e. Submit it. In the FSVIEW window, there should be one observation for each full-duplex interface and no other observations.

- f. Close the FSVIEW window.

The SAS informat CSFDX is used by the IT Service Vision process step when it calculates ifUtilization, so for the utilization calculation to be done properly you must run the above statements *before* you run the process step.

If you add or change full-duplex interfaces, you will need to run these statements again. The new information will be utilized the next time that you run the process step.

- b. **Check the SAS log for error messages.**

- c. **Decide whether the modifications are working to your satisfaction.**

If not, continue to customize the PDB by cycling through the actions in this task until you are satisfied with the results.

Section 2, Task 2: Create Your Production PDB

You now have the test PDB customized the way that you want it. This task will walk you through creating your production PDB and copying the structure and settings and data of your test PDB to your production PDB.

Prerequisites

1. **Your test PDB is the active PDB.**
2. **You have write access to your test PDB.**
3. **You have chosen a name for the PDB that this task will produce.**

In this task you will create a permanent PDB for your collector's data. In this document, we will refer to that PDB as your production PDB.

A typical name is something like:

Collector	PDB Location
ACC	(<i>directories</i>)/prodacc-pdb
MWA	(<i>directories</i>)\prodmwa-pdb

where *directories* is the directory path that you want to use for your production PDB.

Actions

1. **Estimate the size of your production PDB.**

Use an operating system utility such as `du -s` (for UNIX) or Windows Explorer (for Windows NT) to check for the amount of space used by each library (DETAIL, DAY, WEEK, MONTH, YEAR) of your PDB. This amount indicates the amount needed for one unit of data (where a unit is a day for the DETAIL library and the DAY library, a week for the WEEK library, a month for the MONTH library, and a year for the YEAR library). Extrapolate for the length of time (age limit) that you selected for each level.

The amount of data at the day, week, month, and year reduction levels is a function of the number of unique combinations of class variables values that are present in your data.

Note: The amount of space used at week, month, and year levels does not show any appreciable increase until data for another week, month, or year are processed. The data currently stored in the week level, for example, represent a summarization of the whole week and, as such, will be modified by updates to the data during the week, not by additions.

2. **Decide where to create your production PDB.**

Your estimate of the size of the fully populated PDB may affect where you want the production PDB to reside.

3. **Create your production PDB.**

- a. **Create an empty production PDB.**

From the main window in IT Service Vision, follow this path:

Administration -> Create PDB - Wizard

Provide the name of the new PDB. Then select **Next** in every window and **Finish** in the wizard's final window.

b. Copy your test PDB's structure, settings, and data to your production PDB.

Follow this path from the main window:

Administration -> Manage PDBs -> *select PDB to copy* -> *right mouse click* -> copy

If the **From** field does not have the location of your test PDB, select the arrow to the right of the **From** field. In the **Copy To PDB** window, scroll to the name of your test PDB, and select your test PDB. Select **OK**.

Select **OK**. After IT Service Vision completes the copy, you should get a message at the top of your window that says

Note: All tables are successfully copied.

To return to the list of known PDBs, select **OK**. The list of known PDBs is updated to reflect the new PDB.

Select **OK** to return to the main window.

Section 2, Task 3: Customize Your Report Definitions

In this task, you will customize the report definitions that you used in Section 1.

Prerequisites

1. **Your production PDB is the active PDB.**

Administration -> manage PDBs -> *select the production PDB* -> Activate -> WRITE -> OK -> OK

2. **You have decided on storage locations for your customized report definitions.**

Report definitions exist in two forms, interactive and batch/background.

The interactive form is ready to run in the IT Service Vision GUI. This form is, in effect, a list of the parameters values that are to be inserted in the report definition window. Another name for the interactive form is the *report* (window) *definition*.

If you run that report definition from the report definition window, IT Service Vision creates the source code that invokes the appropriate report macro with those parameters and executes that macro invocation. This macro invocation is the batch/background form of the report definition, and another name for it is the *source* (code for invoking the report) *macro*.

In this task you will save

- the interactive form of your report definitions to the ADMIN library of your PDB.

To make use of that library, you do not need to do anything now. Later, you will select that library when asked.

- the background form of your report definition (that is, the source macro) to a flat file.

Actions

1. **Display one of the report definitions that you used in Section 1.**

From the main window, follow this path:

Reporting -> Manage Report Definitions -> *from the Folders list, select the appropriate folder* -> *from the Reports list, select one of the report definitions that you used in Section 1* -> right click on it -> Modify

2. **Customize the title generated by that report definition.**

- a. **Open the Set Title and Footnotes window.**

From the Manage Reports Definition window, follow this path:

Locals -> Set Titles

- b. **Add your title(s).**

Type in your titles in the spaces provided for titles.

Note: You can also add footnotes.

c. Exit the Set Titles and Footnotes window.

Select **OK**.

3. Save the customized report definition in interactive form.

From the Manage Report Definition window, follow this path

File -> Save Report Definition

IT Service Vision displays the Save Report Definition window.

If the original report definition was supplied, the Name and Description are copied from the supplied report definition. To identify the custom report definition uniquely, change at least one character in the the name. For naming conventions, see [Shared Appendix 7: Supplied Reports Naming Convention](#), which includes information about naming conventions for user-created report definitions.

If the original report definition was not supplied, fill in the Name and Description; for example, the name **Faxexamp** and the description **Fax example**.

Select folder **ADMIN.ITSVRPTS**.

Then select **OK**. IT Service Vision creates the custom report definition in the **ITSVRPTS** catalog in the **ADMIN** library of your PDB and returns you to the Manage Report Definitions window.

Select **Close** to return to the IT Service Vision main window.

4. Save the customized report definition in batch/background form.

a. Modify the report definition so that it sends the report to a catalog entry instead of to a monitor display.

Follow this path:

Manage Report Definitions -> Locals -> Report Output Options -> SAS Catalog

In the Output Name field, type the report name. In the Description field, type the report description. In the Location field, type `ADMIN.report_name`. Then select **OK**.

Where `report_name` is the name of your report.

b. Then write out the report definition by following this path from the report definition's main window:

Locals -> Preview Macro Source

IT Service Vision displays the PREVIEW window. This window contains the source statements that generate the report.

Save the modified report definition to a file by selecting **File, Save as, and Write to a File**.

IT Service Vision displays the Save As - PREVIEW window. Adjust the **Save in** field to the name of a directory that you can use to save the report definition. (In the next task, when you are setting up your daily production job, you will refer to this file.) Adjust the **File name** field to the name of the report definition. Then select **Save**.

IT Service Vision writes the report definition in the form that is for batch use (and with the parameters that will direct a generated report to the ADMIN library).

c. To return to the Reporting tab, select **File, End, and Close.**

5. Repeat these actions for the other report definitions that you used in Section 1.

Section 2, Task 4: Set Up Your Production Job

This task walks you through setting up and scheduling the batch or background job that will automatically process, reduce, and report on your site's data.

Prerequisites

1. **You understand how to use the job scheduling system at your site.**

Refer to the documentation for your task scheduler for more information.

2. **You understand the macros for return code checking.**

See the Macro Reference documentation for more information on return code checking.

3. **You have IT Service Vision prototypes and your collector's prototypes for the production job.**



The prerequisite files for moving your PDB into production status are:

a. **the shell script prototypes that IT Service Vision supplies.**

There are examples of batch processing scripts for your IT Service Vision PDB in the [IT Service Vision Showroom](#). These scripts automate the logging of data from your collector as well as the processing of and reporting on the data.

There are complete instructions in the Showroom for automating the logging, processing, and reporting on data from the following collectors:



There are simple examples of batch processing scripts that automate the logging of data from the following collectors in the [IT Service Vision Showroom Script Examples](#).

There is also an example script that automates the running of these scripts on a daily, weekly, or monthly basis in the [IT Service Vision Showroom Script Examples](#).

To create batch scripts that automate the logging of data from other collectors, see your collector vendor's documentation.

Note: Any mention of the *pr_script* later in this document refers to the processing and reducing script for your collector.

b. **the batch/background files that you generated in [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Data](#) and revised in [Section 2, Task 3: Customize Your Report Definitions](#).**

i. **Edit the start, process, and reduce code into a single file in the following order:**

- The code that starts IT Service Vision and processes the data into the PDB.
- The code that reduces the data in the PDB.

This combined file is your collector's version of the IT Service Vision prototype *pr_script* file.

ii. **Edit the code for generating the reports into a single file in any order.**

This combined file is your collector's version of the reporting macros that are included in the IT Service Vision prototype file named *csdaily*. Note that *csdaily* also includes other statements that you will need.



The prerequisite files for moving your PDB into production status are:

a. **the shell script prototypes that IT Service Vision supplies.**

There are examples of batch processing scripts for your IT Service Vision PDB in the [IT Service Vision Showroom](#). These scripts automate the logging of data from your collector as well as the processing and reporting on the data in your PDB.

There are complete instructions in the Showroom for automating the logging, processing, and reporting on data from the following collector:



To create batch scripts that automate the logging of data from other collectors, see your collector vendor's documentation.

Note: Any mention of the *pr_script* later in this document refers to the processing and reducing script for your collector.

b. **the batch/background files that you generated in [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Data](#).**

i. **Edit the start, process, and reduce code into a single file in the following order:**

- the code that starts IT Service Vision and processes the data into the PDB.

- the code that reduces the data in the PDB.

This combined file is your collector's version of the *pr_script* file.

ii. **Edit the code for generating the reports into a single file in any order.**

This combined file is your collector's version of the reporting macros that are included in the IT Service Vision prototype file named *csdaily*. Note that *csdaily* also includes other statements that you will need.

Actions

1. **Create the production job(s).**

Creating your production jobs:

Make a copy of the example (IT Service Vision prototype) scripts that you located earlier and edit them to work in your environment. Follow the instructional comments embedded in the example scripts.

If there is no example script for your particular data source, the example scripts for other data sources will probably still be helpful by suggesting to you ways of automating the processing of your data.

2. **Schedule the production job(s).**



a. **For Cabletron SPECTRUM, schedule Spectrum Data Export to occur automatically.**

- i. Start SDE from within SpectroGRAPH by selecting

File -> Export Data...

- ii. To open the Export Definition Description that you created earlier for exporting the archived data, select

File -> Open...

Select the name of the export Definition Description that you created in [Section 1 Task 2: Start Your Data Collection Software](#). The suggested name was `exp_def_desc_1.sde`. Review the settings to make sure everything is correct for batch processing. For instance, the model data set and statistics data set should be marked for export and the range should probably be a relative specification like `DAY` rather than `custom`.

- iii. To set up regular export intervals, select

File -> Schedule Export...

If you need more information on scheduling automatic data export, see the *SPECTRUM Data Export User's Guide*.

b. Edit your crontab file to execute your scripts that process, reduce and report on the data.

i. Retrieve your current cron file.

You will be adding jobs to your crontab file. Retrieve your current crontab file with the following command:

```
crontab -l > my_crontab
```

ii. Edit *my_crontab* file.

Add the following lines to *my_crontab*:

```
# Backup your pdb at 00:04 every night
04 00 * * * cp -r /u/me/pdb-production /u/me/pdb-backup 2>&1
| mail your_userid
# Run CxPROCES and CPREDUCE at 00:05 every night
05 00 * * * /tmp/pr_script 2>&1 | mail your_userid
# Daily reports each morning at 03:05
05 03 * * * /tmp/csbatrep -d 2>&1 | mail your_userid
# Weekly reports each Sunday morning at 03:20
20 03 * * 0 /tmp/csbatrep -w 2>&1 | mail your_userid
# Monthly reports each first day of month at 03:35
35 03 1 * * /tmp/csbatrep -m 2>&1 | mail your_userid
```

where *your_userid* is your e-mail address, *pr_script* is the script that processes and reduces raw data into your PDB, and *csbatrep* runs the reports on the data in the PDB.

iii. Replace your current cron file with the one that you just edited.

Issue the following command:

```
crontab my_crontab
```



Follow the documentation for your scheduling software to automate the running of your production batch scripts that process, reduce, and report on your data.

Section 2, Task 5: Prepare for Client Use of PDB(s)

This task readies the IT Service Vision server for use with the IT Service Vision client. You can skip this task if your site does not plan to use the IT Service Vision client.

Note: The IT Service Vision client can use remote access to run "directly" on the data in the PDB(s). Or the IT Service Vision client can run on a downloadable subset of data from the PDB(s). You can prepare for either or both of these options.

Similarly, the use of these two options on the client host is independent. The client GUI (except for DeskTop Reporter) runs against the data in the PDB(s). The DeskTop Reporter component of the client GUI runs against a downloaded subset of data from the PDB(s).

Note: The DeskTop Reporter is a SAS/EIS application that runs against IT Service Vision data.

Prerequisites

1. None.

Actions

1. **For remote access to data on one or more PDBs from client hosts, make an e-mailable file of remote profile information.**
 - a. People who want to access a PDB on the IT Service Vision server host from a remote IT Service Vision client host will need information from you to fill out a remote profile. You may want to put the following required information in a file so that it is easy to respond to their requests:
 - i. the name of the server host
 - ii. the name of the operating system on the server host

The valid values are: **MVS** or **UNIX** or **Windows** and **OS/2**.
 - iii. the type of server

The valid values are **CONNECT** and **SHARE**. Use **CONNECT** unless you have set up a **SHARE** server.
 - iv. the communications method

This will be **TCP** in most sites.
 - v. the location of IT Service Vision software on the remote server

The location of IT Service Vision software on the remote server is not required, but is highly recommended.
 - vi. the remote login profile to use

The default login script will suffice in most cases.
 - vii. All other items are optional.

For more information on *all* of the fields in a remote profile, follow this path:

From the main window, select **Administration -> Manage PDBs -> File -> Include PDB -> click on the rightward arrow that is associated with the Remote Server Profile field -> UNIX or MVS-> OK-> Help**

Note: Because IT Service Vision clients usually have network access to the server if the server is on Windows NT, the clients rarely need to use a remote profile to access a PDB on a Windows NT server.

2. For downloading PDB data to client hosts, make a SAS library containing a subset of the data in the PDB(s).

The following instructions prepare a downloadable SAS library of data for use by the DeskTop Reporter component of the client GUI. The macro that creates the downloadable library should be added to your daily production job after the process and reduce steps.

Note: The instructions apply primarily to RMF data from a PDB on the MVS server and to HP MeasureWare data from a PDB on the Open Systems server.

The summarized data used by the Open Systems Data DeskTop Reporter are created using programs and scripts from the IT Service Vision Showroom. The Showroom provides examples of exploiting the facilities of IT Service Vision to organize and report on information technology data sources using a variety of data presentation techniques.

These techniques include using interactive "drill-down" reports in the DeskTop Reporter as well as html-based reports that are built automatically from your data. The Showroom appears on the same CD as the DeskTop Reporter. To browse the contents of the Showroom, point a frame-capable browser (such as Netscape 3.x or Microsoft's Internet Explorer 3.x) to the welcome.htm file in the SHOWROOM directory on the CD and follow the instructions there for the DeskTop Reporter.

The DeskTop Reporter can be used as-is to report on your PDB data, but it is most valuable to you when you extend it to suit your own needs.

Modifying the DeskTop Reporter consists of three phases:

1. Copy the DeskTop Reporter programs and data to your own, writable area.
2. Remove the supplied reports you do not want.
3. Update the list of data sets to download to remove the data sets that correspond to the reports you removed in step 2 above.
4. Modify the remaining reports to suit your taste.

1. COPY THE DESKTOP REPORTER

To create your own copy of the Desktop Reporter, see "IT Service Vision Client CD: Installation and Usage" included in printed form with the Desktop Reporter software shipment or point a Web browser to the Showroom file:

`showroom/html/instusag.htm`

on the CD containing the Desktop Reporter and Showroom software.

After copying the portions of the Desktop Reporter of interest to you,

modify your copy of the autoexec.sas file to remove references to the applications you did not copy, change your working directory to the directory containing the Desktop Reporter files, and start SAS and the Desktop Reporter in the normal way.

2.REMOVE UNNEEDED SUPPLIED REPORTS

To remove unneeded reports, you will need to have licensed either IT Service Vision or SAS/AF. Start the SAS system and enter the EIS command in the command area of the tool bar. Select "Build EIS" and set the "Path" to your copy of the OSYSEIS catalog and the "Application Database" to "Open SystemsDeskTop Reporter". Identify the reports and buttons you do not want in "STRUCTURE OF OPEN SYSTEMS DESKTOP REPORTER" below and remove their entries from the BLOCK or LISTMENU that invokes them (you are removing only the pointer to the EIS object -- not the object itself -- they are not very big and can be useful as examples later on).

To remove an entry from a LISTMENU or BLOCK menu, select and "Edit" the menu entry and select "Edit menu text". Use the vertical scroll bar to position yourself to the entry to be removed and select "Delete".

3.UPDATE THE LIST OF DATA SETS TO DOWNLOAD

Downloading application data from Unix is controlled by a list of data sets in SAS data set DTR_DOWN. This data sets exists in each application's EIS library. For example, the list of data sets to be downloaded for OSYS data is in OSYSEIS.DTR_DOWN.

To avoid attempting to download data sets you do not want for the reports you have just removed or to add new data sets for reports you create, update the download data set list. See "UPDATING THE LIST OF DATA SETS TO DOWNLOAD" in Help entry "Downloading Data from Unix" for instructions on how to update the list.

4.MODIFY THE REMAINING REPORTS

To modify one of the supplied reports, start SAS and EIS and select "Build EIS". From the Build EIS screen, select the entry for the report you want to modify, "Copy" it to a name of your choice, and then "Edit" the copy.

To cause data sets required by the updated reports to be downloaded, you will need to update the list of data sets to download. See "UPDATING THE LIST OF DATA SETS TO DOWNLOAD" in Help entry "Downloading Data from Unix" for instructions on how to update the list.

You can use the Help button associated with the Edit screen for the type of entry you are updating for suggestions of how to change the report. For an overview of what other things are possible with SAS/EIS, select the "Tutorial" on the EIS Main Menu. See also publications "Getting Started with SAS/EIS" and "SAS/EIS Reference" available from SAS Institute Book

Sales.

STRUCTURE OF OPEN SYSTEMS DESKTOP REPORTER

The following is a concise representation of the menu structure of the Open Systems DeskTop Reporter. This list can be used to identify the names of components that produce a report so as to modify or remove references to the component.

Syntax of each line is:

EntryName:InputDataName - TextDescription

where

EntryName name of the catalog entry or EIS object

InputDataName name of summary ds or mddb in OSYSDATA

TextDescription text description of the entry

EntryName uses these abbreviations:

o. = osyseis.osyseis

i. = itsveis.itsveis

InputDataName uses these abbreviations:

.m = mddb

.s = summary data set

```
o.main.block                - Open Systems Application menu

o.cpuservs.listmenu         - CPU Servers Reports
o.srvglb02.multcol:srvglb.d - CPU Servers Basic Metrics
o.srvglb01.multcol:srvglb.d - CPU Servers Extended Metrics
o.srvglb03.groupbar:srvglb.d - CPU Servers CPU Util
o.srvglbq1.groupbar:srvglb.d - CPU Servers Congestion
o.srvglbd1.groupbar:srvglb.d - CPU Servers Disk I/O
o.srvglb50.graphs:srvglb.m  - CPU Servers CPU Utilization

o.s10servs.listmenu         - Busiest Servers Reports
o.s10glb02.multcol:s10glb.d - Busiest Servers Basic Metrics
o.s10glb01.multcol:s10glb.d - Busiest Servers Extended Metrics
o.s10glb03.groupbar:s10glb.d - Busiest Servers CPU Util
o.s10glbq1.groupbar:s10glb.d - Busiest Servers Congestion
o.s10glbd1.groupbar:s10glb.d - Busiest Servers Disk I/O
o.s10glb50.graphs:srvglb.m  - Busiest Servers Utilization

o.workstns.listmenu         - Workstation Reports
o.w15glb01.multcol:w15glbd.d - Busiest Workstations
o.l15glb01.multcol:l15glbd.d - Least Busy Workstations
o.walglbx1.multcol:walglbd.d - Explore Workstations
o.wkspec01.multcol:wkspec.d  - Workstation Network Traffic

o.network.listmenu          - Network Reports
o.wkspec01.multcol:wkspec.d  - Workstation Network Traffic
```

o.download.listmenu	- Download Open Systems Data
o.signunix.af	- Signon to UNIX
o.signunix.scl	- Setup download from UNIX
i.signon.program	- Signon to UNIX
o.downdata.af	- Download Data from UNIX
o.downdata.scl	- Set applid=OSYS for download
i.download.scl	- Download data from remote host
i.downprof.scl	- Create SASUSER.DTR_PROF
i.downfrom.program	- Ask for path to download from
i.signoff.af	- Signoff from Remote Host
i.signoff.scl	
i.downunix.af	- Help for downloading data
o.osyshelp.listmenu	- Help
i.using.af	- Using the DeskTop Reporter
i.downunix.af	- Help for downloading data
o.modify.af	- Modifying the DeskTop Reporter

People who want to download the SAS library will need:

- i. the name of the downloadable SAS library
- ii. the location of the downloadable SAS library.

3. People who want to use the IT Service Vision client will need the following access from you.

- i. read access to the PDB(s), SITELIB, and PGMLIB on the server and/or read access to the downloadable SAS library on the server.

Section 2, Task 6: Wrap Up

This task walks you through the steps necessary to make the production PDB and the production reports available to IT Service Vision users. It also walks you through other steps that complete the transition into production mode.

Prerequisites

1. You know the complete names of the production PDB, SITELIB, PGMLIB, and platform-specific structures.

a. If you are in doubt about the complete names of the production PDB, SITELIB, and PGMLIB, you can:

- invoke the IT Service Vision server
- check that the production PDB is the active PDB
- at the command line or command windows, issue the LIBNAME command
- for the SITELIB and PGMLIB, the complete names are listed in the right-most column
- for the production PDB, look up the complete names for DETAIL, DAY, WEEK, MONTH, YEAR, etc. The common part of these names is the complete name of the production PDB.

b. If you are in doubt about the complete names of the platform-specific structures,

You will need to know the complete name of the `misc` directory

UNIX `!sasroot/misc/cpe`

WNT `!sasroot\cpe\sasmisc`

2. You have a (non IT Service Vision) reminder file.

Another name for a reminder file is a tickler file.

Actions

1. Using your site's security system, verify (and adjust if necessary) access rights for IT Service Vision structures.

The IT Service Vision administrator and general users require different access rights from the file and security system.

Administrator User		
PDB	read/write	read
SITELIB	read/write	read
PGMLIB	read/write	read

Platform specific structures:

UNIX		Administrator	User
	misc/cpe	read	--
WNT		Administrator	User
	cpe\sasmisc	read	--

Note: For a summary of which activities require which access rights to which structures, see [Shared Appendix 3: Actions and Access Rights Required](#).

2. Add items for recurrent tasks to your reminder file.

- Update your site's holiday list in SITELIB once a year.

For details, see [Section 1, Task 1: Start the IT Service Vision Server Software](#).

- Update your site's work shift grid at seasonal shift changes.

For instance, some sites switch to a summer schedule at the beginning of the summer and return to the regular schedule at the end of the summer.

For details, see [Section 1, Task 1: Start the IT Service Vision Server Software](#).

- For the PDB, periodically check the space use against the space available, and move the PDB if necessary.
- Periodically review manually entered network-device configuration information.

For details, see [Section 2, Task 1: Customize and Verify Your Test PDB](#).

- When you change interfaces, check the format for full-duplex interfaces.

For details, see [Section 2, Task 1: Customize and Verify Your Test PDB](#).

3. Congratulations! You have successfully set up for production.

For a list of other documentation, see the Roadmap under IT Service Vision **OnlineHelp -> Documentation**.

For example, you may want to see

- the Showroom for
 - how to web-enable the production reports
 - other suggestions for working with your collector and other collectors
- the Help documentation for
 - more about reporting
 - more about the client
- the Macro Reference documentation for
 - more about the parameters on your macro invocations
 - more about the data dictionary utility.
- Section 3 of the Setup doc for (optional) advanced setup tips.

Section 3, Advanced Setup

Section 3: Advanced Setup

You may want to consider:

1. Deleting one or more demonstration PDBs.

At some sites, copies of the demonstration PDBs are used for practice or testing; for instance, testing what a new report would look like when the production PDB does not have enough data to generate a report that looks good. At some sites a demonstration PDB is used when the active PDB must be some other PDB than the one that is to be used. (Some operations cannot run on the active PDB.) At other sites, one or more of the demonstration PDBs are never used. You can delete one or more of these PDBs if you have no need for them.

In case you change your mind later, back up the PDB that you will delete.

To delete the PDB, follow this path from the IT Service Vision main menu:

Administration -> Manage PDBs -> select the PDB -> right mouse click -> Delete PDB -> Yes -> OK

2. Combining one or more PDBs.

You may have one or more PDBs for data from one collector and also one or more PDBs for data from other collectors.

There are advantages to having data from different collectors in *different* PDBs:

- You can back up, process, reduce, and report on the data in parallel.
- You are less likely to run into file size problems (UNIX has a 2-gigabyte filesize limitation).
- It may be easier to handle sites with different time zones.
- It may be easier to handle multiple users needing update access at the same time.
- It may be easier to manage space.

There are also advantages to having data from different collectors in the *same* PDB:

- If you want to generate a report on data in more than one table, it is more convenient to have the tables in the same PDB.
- If you want to generate reports interactively, you can work with one PDB.

You can copy tables (including the data in the tables) into a combined PDB at any time that using the same PDB would be convenient. You can use the same method to copy tables (including the data) into separate PDBs at any time that using different PDBs would be convenient.

To combine tables into one PDB:

a. Decide which PDB to use for the combined PDB.

You can use an existing PDB or a new one. If you decide to use a new one, you can create it (without tables) by following this path from the main menu:

Administration -> Create PDB - Wizard -> follow wizard except omit table selection

Note For UNIX and WNT, do not select tables to add. Do request that the new empty PDB become the active PDB.

b. Change to another PDB as active PDB.

To list the known PDBs, follow this path from the main menu:

Administration -> Manage PDBs

IT Service Vision displays your list of known PDBs with the name of the active PDB in the header.

Switch to any other PDB than the PDB that will contain the combination of tables (for instance, switch to a demonstration PDB) as the active PDB.

select any other PDB -> Activate -> OK

IT Service Vision re-displays your list of known PDBs and, in the header, names the selected PDB as the active PDB.

c. Copy tables (including data) to the combined PDB.

For each PDB that has tables that are to be copied to the combined PDB, follow this path from the list of known PDBs:

select the PDB -> right mouse click -> Copy -> specify copy from a PDB and copy to PDB -> OK

Note: These steps are not capable of or suitable for combining the observations from identically-named tables.

d. Check that the tables and data are in the combined PDB.

On the list of known PDBs, activate the combined PDB

select the combined PDB -> Activate -> OK

Then examine the data:

Administration -> Examine PDB Data

The tables that you copied should be listed, along with a count of the number of observations in the specified level.

If you want to see the data, **select a table with non-zero observations-> right mouse click -> View Data.**

When you finish looking at the data, **Close -> OK.**

e. Change your batch/background job(s) to use the combined PDB.

See [Section 2, Task 4: Set Up Your Production Job.](#)

Additional Advanced Setup Topics

Consider creating new report definitions.



Decide if you want to use any of the other HN2 tables and, if so, which other variables you want to use from those tables.



Consider collecting SNMP data using the other MIBs for which IT Service Vision supplies tables.



Consider collecting SNMP data using MIBs for which IT Service Vision does not supply tables.

Consider creating new report definitions

In this example, you will create a new, custom report definition.. You will also use formula variables for customizing report definitions. You must have at least two days of data to get a good report.



Read "Using One or More Formula Variables" in OpenView and NetView Appendix 3: OVW-Specific Tips for Reporting to learn about formula variables. (The example below uses table HN2NIX. You do not need to create the formula variables UDAY and UWEEK in HN2NIX. It happens to be the case that they are supplied with this table. It would, however, be useful to understand how they are constructed, because the same method would work for your own formula variables.)



Read "Using One or More Formula Variables" in Cabletron SPECTRUM Appendix 3: SPECTRUM Specific Tips for Reporting and then create at least the formula variables UDAY and UWEEK.

Here is an example of a new report definition that makes use of one or more formula variables:

- a. Follow this path from the IT Service Vision main window:

Reporting -> Manage Report Definitions

IT Service Vision displays the main report definition window. (For details about this window, select **OnlineHelp -> Help on this window.**)

- b. To create a new report definition, select **File -> New Report Definition**.
- c. To select the table whose data is to be used in the report definition, select the arrow to the right of the Select Table field, and then follow this path, starting at All:

CSi All -> DAY.CSIMRX -> OK

OVNV All -> DAY.HN2NIX -> OK

OV All -> DAY.HN2NIX -> OK

- d. To select the variables whose data is to be used in the report definition, select the arrow to the right of the Select Variable field, and then select **Analysis -> IPIDELV Mean -> Add -> X Variable -> HOUR -> Add -> Class -> UDAY -> Add -> By -> MACHINE -> Add**.

Check that you have selected these variables by selecting **Show All**. You should see:

Variable	Variable Type	Description
IPIDELV	ANALYSIS	MEAN: # datagrams delivered/sec
HOUR	XVAR	Hour
UDAY	CLASS	FORMULA: Day
MACHINE	BY	Machine

(If you need to correct the list, select the variable type of the incorrect variable, select the incorrect variable, and select **Remove**. Then select the correct variable.)

Select **OK** to return to the main report definition window.

- e. To define which subset of the data is to be used by the report definition, select the arrow that is to the right of the Subset Data field and then select **Query/Where Clause Builder**. Using the buttons that are to the left of the Local Query/Where Clause For This Report field, select **Variable -> MACHINE -> OK -> Operator -> IN -> OK -> Value -> with the Ctrl key pressed, select the servers on which you want to report -> OK**.

Select **OK** to return to the main report definition window.

- f. To specify the summary time period, select the arrow to the right of the Summary Time Period Field. Then select **As Is -> OK**.
- g. To specify the report style, select the image in the Current Report Style field. (That is, select the image below the words Current Report Style.) Then select **3D Graphs -> Pillar -> OK**.
- h. To run the report definition and generate the report, select **Run**. You should see a 3D "cityscape," with the Hourly Mean of the Number of Datagrams delivered per second on the Y axis, the hour of the day on the X axis, and the day of the week on the Z axis. This report is for the last machine that you specified in the Where Clause. To see the reports for the other machines, select **View -> Backward**.
- i. To return to the main report definition window, select **File -> End**.

If you have at least several weeks of data in your PDB, you can also experiment with a variation on this report definition: follow the same steps but use UWEEK where you used HOUR.

If you find this kind of report useful, you can save reports and report definitions from the main report definition window by selecting **File -> Save Report Definition**. in the same way that you did in [Section 2, Task 3: Customize Your Report Definitions](#).

You can also make any number of custom report definitions using **Manage Report Definitions** on the **Reporting** tab. Using UDAY and UWEEK is not necessary. You can use any variables in the specified table and any other features available in the specified type of report definition.

Note: If you make custom report definitions, remember to add them to the appropriate batch/background job file. For details, see [Section 2, Task 3: Customize Your Report Definitions](#).



Decide if you want to use any of the other ^{HN2} tables and, if so, which other variables you want to use from those tables.

Note that these other tables duplicate the HN2NIX table metrics, so you probably do not want both. You may want to use the small tables instead of the HN2NIX table if you want to collect data at different intervals for different tables.



Consider collecting SNMP data using the other MIBs for which IT Service Vision supplies tables.

IT Service Vision provides supplied tables for data from other SNMP MIBs such as the Cisco MIB (which corresponds to the tables whose names start HHC), the RMON (RFC1271) MIB (which corresponds to the tables whose names start HRM), and the Wellfleet and Wellfleet Series 7 MIBs (which correspond to the tables whose names start HWE). To see a list of pre-defined tables for HPOV/NVAIX, follow this path from the main window:

Explore Tables/Variables CPE -> Collector: Selected -> HP-OV -> OK

With one exception, you collect, process, reduce, and report on data from these MIBs in the same way that you handle data from MIB-II. The exception is that MIB-II was loaded into HPOV/NVAIX when HPOV/NVAIX was installed. You will need to load the other MIB or MIBs into HPOV/NVAIX. For load instructions, see HP OpenView Network Node Manager Administrator's Reference or NetView for AIX Administrator's Guide.

Note: For the naming conventions for tables see [Open Systems and Windows NT Appendix 1: Tables and Variables Naming Convention](#).



Consider collecting SNMP data using MIBs for which IT Service Vision does not supply tables.

To construct IT Service Vision tables that correspond to the MIBs for which IT Service Vision does not supply tables, you need to run the IT Service Vision mib2dict program on the MIB. For instructions on running the program, see [Open Systems and Windows NT Appendix 4: MIB to Dictionary Compiler](#).

Platform - Specific Appendices

Open Systems and Windows NT Appendix 1: Tables and Variables Naming Convention

SUPPLIED TABLE NAMING CONVENTION

IT Service Vision supplied tables are named according to the following scheme:

cmmnnn

+-----Name: Abbreviation for the group of metrics in
this table

For example,
ICM - nb-mibII.icmp
NIX - nb-mibII.non-indexed metrics

++-----Collector (three-character abbreviation) or
Collector (one-character abbreviation) plus Creator
two-character abbreviation):

Three-character abbreviations are

ACC - Unix Accounting (ACCTON)
PCS - Performance Collection Software and MeasureWare by HP
SNM - SunNet Mgr RPC-Agents by Sun/Connect
TKR - TRAKKER by Concord Communication
TMN - Landmark TMON for UNIX
PRX - Probe/X by Landmark Systems
Rxx - Rolm PBX
SAP - SAP R/3
NTx - Windows NT Server
WEB - Web Log Server

+-----Creator: (two-character abbreviation)

For example,
N2 - NB MIB-II
WF - Wellfleet MIB

+-----Collector (one-character abbreviation)

C - Spectrum by Cabletron Inc.
H - HP OpenView Node Mgr or IBM NetView for AIX
S - SunNet Manager by Sun/Connect
M - Mib-based Data

HOW TO FIND A SUPPLIED TABLE

Note: If you search by MIB name, you may need to try several versions of the MIB name. A common name of the MIB (like ``MIB-II'' or ``RMON MIB'') may not be in the description. SAS/CPE names MIB tables after the name used in the MIB itself. For example, the opening line of the MIB defined by RFC1271 (also known as the RMON MIB) is

RFC1271-MIB DEFINITIONS

Thus SAS/CPE identifies tables in this MIB with the string RFC1271-MIB. The name of each individual table/group (like ``ICMP'' or ``alarmTable'') is also in the description field.

For example, the description field for table HN2ICM begins
nb-mibII.icmp and for table HRMATM begins
RFC1271-MIB.alarmTable.

USER-DEFINED TABLE NAMING CONVENTION

Begin the names of user-defined tables with the character U. Names
must be exactly six alphanumeric characters.

Open Systems and Windows NT Appendix 2: Tips for Reports

- **A Collection of Tips**
- **Tips for Subsetting Data**
 - Subsetting via DATETIME
 - Subsetting by a Simple Local Where Expression
 - Subsetting by a Simple Global Where Expression

A Collection of Tips

Here are some tips that may be useful to you as you generate reports with the IT Service Vision GUI.

- If running your report definition does not seem to produce any results, be sure that you are looking for the report in the correct window. The default window for graph reports is the SAS GRAPH window. The default window for text reports is the SAS OUTPUT window. You can override those defaults from the report's main definition window by selecting **Output Options, Graphics Window** or **Output Window**, and **OK**.
- If running your report definition produces unexpected results, examine the SAS log for messages.
- Some IT Service Vision supplied report definitions require setup to run properly.

Follow this path from the IT Service Vision main window

**Reporting -> Explore Supplied Reports ->
select the report definition ->
click right mouse button -> Show Report Explanation**

to be certain that you have met all the setup requirements for a report definition.

- If you get the No observations were selected message, check the following:
 - Does the SAS log display the report definition that you ran? Check to see that you are running at the (REDLVL) at which you have data.
 - Check to see that you are running with the date range (BEGIN and END) in which you have data.

For more about date range subsetting, see Subsetting via DATETIME in this appendix.

- Check to see that you are running with a subsetting WHERE expression (Local Where or Global Where) for which you have data. If not, and you are running a supplied report definition that requires setup, the lack of a subsetting WHERE expression could easily be the cause. Check the Explanation for the report definition.

For more about a subsetting WHERE expression, see Subsetting by a Simple Local Where Expression in this appendix and Subsetting by a Simple Global Where Expression in this appendix.

- Check to see that there really are data in that table at that level in that date range for the variables in the report definition. From the IT Service Vision main window, follow this path

**Administration -> Examine PDB Data -> select the level ->
select the table -> click right mouse button ->
View Data**

Note: Use can also use %CPPRINT to browse the data. See the Macro Reference documentation for IT Service Vision for more information.

- If the data that you see in your table using Browse Data or View Data are unexpected, check the table's definition. From the IT Service Vision main window, follow this path

Administration -> Manage Tables ->

select the table -> click right mouse button -> Properties

Check that the Kept status of the table is set to Yes. Then select **Variables** and check that the Kept status of the appropriate variables (***select a variable -> click right mouse button -> Properties***) is set to Yes, and the appropriate levels are set to non-zero durations. For the non-detail levels, check that the appropriate statistics are selected. (For a formula variable, check that the formula is applied to the appropriate level and that the table and variables on which it is based are defined appropriately.)

Note: A formula variable does not have selectable statistics. If you want statistics on a formula variable, you must define the statistics as formula variables at the desired non-detail levels.

- Check that the collector was configured correctly to provide the metrics.
- If you do not get the "No observations" window but you get no report, perhaps all data values are missing. That is, the observation may exist but the value on the observation is set to the missing value code.

At detail level, this can occur if you have not set up data logging properly. At all levels, this can occur for a formula variable if it is not defined properly or if the statistics on which it is based are not selected at the levels specified for the formula variable or if the base variables for the statistics have a Kept status of No.

Note: A formula variable does not have selectable statistics. If you want statistics on a formula variable, you must define the statistics as formula variables at the desired non-detail levels.

- Read the Help file on Report Styles. Select **Reporting -> Manage Report Definitions -> *select the Current Report Style image -> Help***.
- The most probable cause of this message

ERROR: The left vertical axis labeled <varlabel> could not be fit as specified.

is that you have more classes than will fit in the legend on the graph.

A solution to this problem is to increase the height of the GRAPH window. Type the following statement in the body of the PROGRAM EDITOR window

```
goptions device=value vpos=100 ;
```

where VPOS=100 means a vertical height of 100 rows. Then submit it and re-run the report definition.

- Test report definitions on a subset of your data so that the testing proceeds quickly.

For details, see [Tips for Subsetting Data](#) in this appendix.

- Add formula variables (variables that are calculated only when accessed) to simplify reporting.

For details, see [Consider creating new report definitions](#) under UNIX and Windows NT in Section 3 and [OpenView and NetView Appendix 3: OVW Specific Tips for Reporting](#).

Tips for Subsetting Data

Subsetting via DATETIME

If your report displays no observations (you get an empty report) or if your report displays observations in only one part of the date range of the processed or reduced data, you may need to reset the report definition's datetime range.

Follow this path from the report's main report definition window

DateTime Range -> Reset from PDB -> OK

IT Service Vision looks in the PDB's data dictionary to find the earliest and latest DATETIME values at the level that is specified on the main report definition window.

If you want to use less than that range, you can type over the values in the Begin and End fields.

Note: The need to reset may occur when you are designing or modifying report definitions and switch from one table to another or one level to another. The begin and end points of the range are global variables that retain their values until you change them or switch the active PDB.

Subsetting by a Simple Local Query Expression

A Local Where expression applies to a single report definition.

If your report definition generates a report that includes observations in which you are not interested, you can use the Local Where to restrict the observations to the ones in which you are interested. For instance, suppose you have a report definition with a BY variables list that includes MACHINE, and you have many machines but you are only interested in a few of them for this report definition. Rather than get dozens of graphs and ignore all but a few, you can generate just the ones you want to see.

On the main Manage Report Definitions window, select the arrow to the right of the Subset Data field and then select Query/Where Clause Builder. For the Local query, follow this path

```
Varname -> MACHINE ->> Operator ->
IN -> Value(s) ->
select the machines on which you want to report ->
OK
```

This method is not restricted to subsetting by MACHINE. You can subset by any variable that is in the specified level of the specified table. For instance, you can report on only first shift data by using this expression

```
SHIFT = '1'
```

Note: By default, if there is an expression in the Local Where, it overrides the expression, if any, in the Global Where. If you want the report definition to use both the Local Where and the Global Where, insert the words SAME AND in front of the expression in the Local Where.

Subsetting by a Simple Global Query Expression

A Global Where expression applies to all report definitions.

You may want to generate a number of reports with the same subsetting; for this, use a Global Where expression. For instance, you may be having trouble with a few machines and want to generate a number of reports for just those machines. In this case, rather than using the Local Where for each report, you can set the Global Where (on any of the report definitions) and it will apply to all report definitions until you reset it to blank.

On the main Manage Report Definitions window, select the arrow to the right of the Subset Data field and then select Query/Where Clause Builder. For the Global query, follow this path

```
Varname -> MACHINE -> Operator ->
IN -> Value(s) ->
select the machines on which you want to report ->
OK
```

This method is not restricted to subsetting by the variable MACHINE. You can subset by any variable that is in the specified level of the specified table. For instance, you can report on only first shift data by using this expression

```
SHIFT = '1'
```

Note: By default, if there is an expression in the Local Query, the Global Query is ignored (unless the **And** is selected between the Local and Global Queries).

Open Systems and Windows NT Appendix 3: Printing Graphs

- Introduction
- Printing Graphs from a Batch Job
 - Prerequisites
 - Actions

Introduction

This appendix steps you through the task of customizing the printing of graphs interactively and in background.

If the monitors and/or printer/plotters at your site differ from those in the examples in this appendix, you may need to refer to the *SAS Companion for UNIX Environments: Language* documentation for your current release of SAS for information about your equipment. Pages 89-90 (in the Version 6 First Edition) are especially relevant to this topic.

If you prefer to use a method other than the ones described in this appendix, you can see other methods in the *SAS Companion for UNIX Environments* and in SAS Technical Support document TS 246, "PRODUCING HARDCOPY GRAPHICS UNDER UNIX." (In TS 246, the information for SAS Release 6.09 applies to Releases 6.09, 6.11, and 6.12.)

If you have questions, ask your SAS Software Representative, or ask your SAS Software Representative to call SAS Technical Support.

Printing Graphs from a Batch Job

Prerequisites

1. **You know the characteristics of your printer/plotter.**
2. **You know the name of the SAS device driver that can handle your printer/plotter.**

1. In your PROGRAM EDITOR window, type the following starting at the first line in the body of the window:

```
proc gdevice;  
run;
```

2. Submit this program fragment.
3. The GDEVICE window lists all of the SAS graphics device drivers. Find one that matches the characteristics of your printer/plotter and make a note of its name.

If the brief descriptions in the Description field are not sufficient for you to decide on a driver, type **B** over the underscore that is to the left of the driver's name and press ENTER or RETURN. More information displays. To return to the list of device drivers, select **File** and **End**.

Note: If you are in still in doubt about the appropriate device driver to use, from the list of device drivers, follow this button path

Help -> Extended Help

If you need more help selecting a driver, ask your SAS Software Representative to call SAS Technical Support.

4. To close the GDEVICE window, select **File** and **End**.

Actions

1. Ensure that you have a GOPTIONS statement that includes a DEVICE= parameter, and set the DEVICE= parameter to the same device driver that you selected above. (You do not need to specify the TARGETDEVICE= parameter.) For example,

```
goptions device=hplj4si ;
```

2. Then specify settings by means of OUT xxxx parameters on the IT Service Vision reporting macros. For example, to route the graphs to a printer/plotter, you would use these parameters on each of the reporting macros:

```
outmode=catalog  
outloc=work.repts
```

where *work.repts* is a temporary catalog that is automatically deleted at the end of the batch job (because the SAS WORK library is automatically deleted at the end of the batch job).

For instance, here is an example batch job that generates two graphics reports, one for mills and one for natural, and sends them to the catalog and to the printer (using a site-specific print command, which is *nlp* in this example):

```
%cpstart(mode=batch,  
          pdb=/nfs/ark/local/disk1/sasabc/pdb/unix,  
          access=readonly);  
  
filename grafout pipe 'nlp';  
goptions cback=white  
        device=ps  
        gaccess=sasgastd  
        gsflen=80  
        gsfmode=replace  
        gsfname=grafout ;  
  
GOPTIONS reset=title reset=footnote ;  
TITLE1 "CPU and Disk Utilization across Days-Mills";  
%CPPLOT2(PCSGLB, GLBCPTO, CPU Total, GLBDUT, Disk Utilization  
        , REDLVL=DAY  
        , PERIOD=ASIS  
        , BY= MACHINE  
        , WHERE=MACHINE = 'mills'  
        , outmode=catalog  
        , outloc=work.repts  
        , outname=plot);  
  
GOPTIONS reset=title reset=footnote ;  
TITLE1 "CPU and Disk Utilization across Days-Natural";  
%CPPLOT2(PCSGLB, GLBCPTO, CPU Total, GLBDUT, Disk Utilization  
        , REDLVL=DAY  
        , PERIOD=ASIS  
        , BY= MACHINE  
        , WHERE=MACHINE = 'natural'  
        , outmode=catalog  
        , outloc=work.repts  
        , outname=plot);
```

Open Systems and Windows NT Appendix 4: MIB to Dictionary Compiler

- **Introduction**
- **Creating Table and Variable Definitions from a MIB**
 - Prerequisites
 - Actions

Introduction

In the master data dictionary, IT Service Vision supplies table and variable definitions for the data from many collectors. For those data, you can use the supplied table and variable definitions (after adding them from the master data dictionary to your PDB). For other data, you can create the table and variable definitions in your PDB by running the IT Service Vision %CPDDUTL macro with CPDDUTL control statements that define the tables and variables.

You can write the CPDDUTL control statements yourself. But for SNMP data (all of which are described with a MIB) there is an easier way to generate the necessary control statements that add the MIB tables and variables to your PDB. The IT Service Vision `mib2dict` compiler takes as input any SNMP V1 or V2 MIB and produces as output CPDDUTL control statements that can be read directly by the %CPDDUTL macro to define IT Service Vision tables and variables. The `mib2dict` compiler can optionally produce SAS statements for input to the SAS FORMAT procedure so that you can add formats for MIB enumeration types.

As an example, this appendix walks you through the steps needed to create table and variable definitions for MIB-II data. The same steps apply for any MIB. We use MIB-II as an example because of its simplicity and familiarity. If you want to use MIB-II tables in your PDB, you do not actually need to use `mib2dict` because all the MIB-II tables are included in the master data dictionary that is shipped with IT Service Vision. (MIB-II table names begin with M.)

Note: For more information about the %CPDDUTL macro, see the Macro Reference documentation for IT Service Vision.

Note: For more about collecting data using the other MIBs for which IT Service Vision supplies tables, see the UNIX information in [Section 3: Advanced Setup](#).

Note: If you need to know more about the MIB for one of your hardware devices, check the hardware documentation or call the manufacturer of the device. There are proprietary MIBs and public domain, Internet standard MIBs. If the MIB is a proprietary MIB, the hardware installation procedure probably copied it to a directory at your site and/or you can probably find a copy on the manufacturer's web page. If the MIB is a standard MIB, the copy is probably on the web page <http://ds.internic.net/>.

Creating Table and Variable Definitions from a MIB

Prerequisites

1. **The IT Service Vision server Release 1 or later is installed on your SAS host machine.**
2. **The IT Service Vision server software is running, and the PDB to which you want to add the tables is the active PDB.**
3. **You are familiar with the `mib2dict` man page.**

To view the man page, execute the following (assuming that SAS has been installed in `/usr/local/sas`):

```
cd /usr/local/sas/utilities/man/man1
nroff -man mib2dict.1 | more
```

4. **The MIB that you want to compile conforms to SNMPv1 or SNMPv2 standards.**

For the purposes of this exercise, we will use `rfc1213.mib`, which is a file containing the definition of MIB-II. Assuming you have installed SAS and the IT Service Vision server in `/usr/local/sas`, you can find a copy of `rfc1213.mib` in `/usr/local/sas/saspgm/cpe/mibs/`.

Actions

1. Compile the MIB.

In this action we are going to compile `rfc1213.mib` and produce two output files: `rfc1213.ddu` and `rfc1213.sas`. The `.ddu` file will contain the CPDDUTL control statements needed by the `%CPDDUTL` macro. The `.sas` file will contain the PROC FORMAT statements needed to create the formats for enumeration types.

To compile the MIB, execute the following command (assuming that SAS has been installed in `/usr/local/sas` and that your `PATH` includes `/usr/local/sas/utilities/bin/`):

```
mib2dict -i rfc1213.mib -o rfc1213.ddu -f rfc1213.sas -c hp-ov
```

where

- the `-i` option specifies `rfc1213.mib` as the input file to be compiled. If you do not specify `-i`, then `mib2dict` will expect input from stdin.
- the `-o` option specifies `rfc1213.ddu` as the output file to which CPDDUTL control statements are printed. If you do not specify `-o`, CPDDUTL control statements will print to stdout.
- the `-f` option specifies that `rfc1213.sas` is to receive the SAS statements for PROC FORMAT. If you do not specify `-f`, then no PROC FORMAT statements are written to `rfc1213.sas`.

Note: When you specify the `-f` option, any CPDDUTL CREATE VARIABLE statements that are generated by `mib2dict` have a `FORMAT=` parameter that is set to the proper SAS format, as defined in `rfc1213.sas`. In a later action, you will execute `rfc1213.sas`. If you do not execute `rfc1213.sas`, the formats will not exist and you will receive format errors when a value that uses the non-existent format is to be displayed.

- the `-c` option specifies what `hp-ov` as the collector to use. If you do not specify `-c`, the collector will be set to an empty string.

Note: When you specify the `-c` option, any CPDDUTL CREATE TABLE statements that are generated by `mib2dict` have a `COLLECTOR=` parameter that is set to `hp-ov`. In this example we use `hp-ov` because we plan to use HP OpenView Network Node Manager or IBM NetView for AIX to collect the SNMP data. You may use any collector supported by IT Service Vision that can collect SNMP data. Or if you are logging SNMP data in an unsupported format, you could use `-c generic` and use the IT Service Vision generic collector capability to read the raw data.

2. Review, and revise if necessary, the resulting CPDDUTL control statements.

Although `mib2dict` is designed to create CPDDUTL control statements that can be directly processed by the `%CPDDUTL` macro, there are circumstances in which you may want to edit the control statements before you use them:

1. There may be tables or variables that were in the MIB but that you do not want to add to your PDB.

If this is the case, prior to executing the `%CPDDUTL` macro you can simply delete the CPDDUTL control statements that define the tables or variables that you do not want.

To delete an entire table, look for the CREATE TABLE statement and delete everything up to the next CREATE TABLE statement.

To delete a variable from a table, simply delete everything from CREATE VARIABLE up to and including the semicolon that ends the statement.

If you delete a variable from a table but you do not delete the entire table, be sure that the variable you deleted is

not in the table's BYVARS or CLASSVARS lists. In particular, do not delete MACHINE, HOUR, or SHIFT because these variables are always on the BYVARS and CLASSVARS lists. For more about the BY and CLASS variables lists, see [Section 2, Task 1: Customize and Verify Your Test PDB](#).

2. There may be some IT Service Vision table or variable names that you want to change.

Although mib2dict is able to squeeze long MIB table and variable names down to six-character names for the PDB tables and seven-character names for the PDB variables, the resulting names do not always look mnemonic.

So that users of your PDB can read the names more easily, you can modify the PDB table and variable names in the file to which the CPDDUTL control statements were written. For the naming conventions of user-defined tables and variables, see [Open Systems and Windows NT Appendix 1: Tables and Variables Naming Convention](#).

3. Run the %CPDDUTL macro to create the table and variable definitions in your PDB.

In the body of the SAS PROGRAM EDITOR window, type the following:

```
%cpddutl(filename='rfc1213.ddu');  
run;
```

Then if your PROGRAM EDITOR window has pull-down menus, select **Locals** and **Submit**; otherwise, type SUBMIT on the command line and press ENTER or RETURN. %CPDDUTL creates the table and variable definitions in your PDB.

4. Optionally create formats for MIB enumeration types.

If you used the -f rfc1213.sas option when you ran mib2dict, then you need to execute rfc1213.sas. This SAS program executes the SAS FORMAT procedure, which creates the new formats and installs them in the active PDB's DICTLIB.CPFMTS catalog.

If your PROGRAM EDITOR window has pull-down menus, select **File** and **Open**, locate rfc1213.sas, and select **Open**; otherwise, type include rfc1213.sas on the command line and press ENTER or RETURN. Then select **Locals** and **Submit**; or type SUBMIT on the command line and press ENTER or RETURN.

5. You are now ready to process raw data into your new PDB.

In our example we used the mib2dict -c hp-ov option to specify HP-OV as the collector. Thus, to process data (for example, for ifTable in the newly built UIFTBL table), you could submit the following from the PROGRAM EDITOR window:

```
%csproces (/usr/OV/databases/snmpCollect/  
          UIFTBL,  
          collector=HP-OV);
```

Collector - Specific Appendices

Cabletron SPECTRUM Appendix 1: Starting the `processd` Daemon

The `processd` daemon is not running on the specified SpectroSERVER if you start SpectroSERVER and get a message similar to this one:

```
SpectroSERVER: Cannot connect to the processd at this location:
machine name
```

where *machine name* is the SPECTRUM management host name.

To start the daemon, perform the following steps on the SPECTRUM management host machine.

1. Check to see if the `processd` daemon is running.

Check to see if the daemon is currently running by issuing this command:

```
ps -ef | grep processd
```

If the daemon is running, you will see output similar to the following:

```
root 975 295 2 14:50:39 pts/1 0:00 grep processd
root 960 1 42 14:48:13 ? 0:00 /usr/spectrum/SDPM/processd -dir
/usr/spectrum/SDPM
```

2. Start the `processd` daemon.

To start the daemon, issue these commands:

```
cd SPECTRUM_home/SDPM ./processd.sh start
```

To see if the daemon has started, repeat Step 1 in this appendix. If the daemon is still not running, check the *SPECTRUM System Administrator's Guide* for troubleshooting hints.

Cabletron SPECTRUM Appendix 2: Alternate Data-Logging Method

This is an alternate version of the task in [Section 1, Task 3: Create a Test PDB and Process, Reduce and Report on Your Data](#). In this version of the task, you update the SPECTRUM database using the SPECTRUM Command Line Interface (CLI). Using CLI, you can enable logging of attributes for more than one model at a time.

Prerequisites

1. **SAS is installed.**

See [Section 1, Task 2: Start Your Data Collection Software](#).

2. **The IT Service Vision server software is installed.**

See [Section 1, Task 2: Start Your Data Collection Software](#).

3. **You know the model type name of the devices from which you want to collect data.**

Because SPECTRUM and IT Service Vision are organized based on the model type of a device, in this document logging will be enabled for all devices with the same model type. See [Section 1, Task 3: Create a Test PDB and Process, Reduce and Report on Your Data](#) for more information on determining the model type for a particular model.

4. **The IT Service Vision server is available on the SPECTRUM management machine.**

See [Section 1, Task 2: Start Your Data Collection Software](#) for more information.

Actions

All actions for this task are to be performed on the SPECTRUM management machine.

Note: The IT Service Vision file `cscsilog` contains a shell script that interfaces with the SPECTRUM data base by using the SPECTRUM Command Line Interface (CLI). For more information on CLI, please refer to the *SPECTRUM Command Line Interface* manual.

1. **Customize the `cscsilog` shell script for updating the SPECTRUM data base to enable data collection.**

- a. **Make a copy of `cscsilog`.**

For the purpose of illustration, assume that `!SASROOT=/usr/local/sas/`, SPECTRUM is installed in `/usr/spectrum`, and assume that your work space is `/tmp/`. In that case, you would use this command:

```
cp /usr/local/sas/misc/cpe/cscsilog /tmp/cscsilog
```

to copy `cscsilog` to the `/tmp` directory.

- b. **Customize the `cscsilog` shell script.**

Near the top of `cscsilog` you will find a section of user-definable variables that control the execution of the script. These must be customized to work properly at your site.

- `DATA_PATH`

Set this to the SPECTRUM base directory to which *SPECTRUM_home* refers. For example, if SPECTRUM is installed in /usr/spectrum, DATA_PATH='/usr/spectrum'.

- DATA_DIR

Set this to the directory (under the base directory) where the command line interface programs are located. Normally, DATA_DIR='vnmsh'.

- POLL_INT

Set this to the number of seconds between polls of the devices. A smaller number will create more network traffic.

- LOG_RATIO

Set this to the fraction of polls that are logged. (For instance, if LOG_RATIO is 5, every fifth poll is logged.) This, together with POLL_INT, controls the intervals between data samples.

2. Execute the script

The script requires at least two parameters. The first is the SPECTRUM landscape (server) name. The remaining parameters are the model types of interest. Many different model types can be enabled for polling and logging at one time. For example, if the SPECTRUM landscape name is sunfin and the model types for data collection are HubCSIMRXi and Rtr_Cisco, then the following should be used to start data collection:

```
/usr/spectrum/vnmsh/cscsilog sunfin HubCSIMRXi Rtr_Cisco
```

3. Watch the output of the script.

You should see messages similar to the following:

```
Starting CLI Local Server ./VnmShd ...
connect: successful sunfin
current landscape is 0x400000
Updating attributes for all models of type Rtr_Cisco
Id      Name      Value
0x10071 Polling_Interval 60
Id      Name      Value
0x10072 Poll_Log_Ratio   10

Updating attributes for all models of type HubCSIMRXi
Id      Name      Value
0x10071 Polling_Interval 60
Id      Name      Value
0x10072 Poll_Log_Ratio   10
```

where the Id fields are the attribute IDs that correspond to the Polling Interval and Poll_Log_Ratio. These attributes control the rate of polling and archiving of data for the model type.

Cabletron SPECTRUM Appendix 3: SPECTRUM Specific Tips for Reporting

Using One or More Formula Variables

Formula variables are variables whose values are calculated (from the values of other variables) when you access the data. The values of the formula variables do not occupy disk space in the PDB.

You can add formula variables to PDB tables by using the IT Service Vision server's interactive interface or by using the IT Service Vision %CPDDUTL macro. It is generally preferable to use the %CPDDUTL macro for your production PDB because the macro documents what you did exactly and enables you to add the variables to many tables easily and accurately.

For information on adding a formula variable using %CPDDUTL, see the Macro Reference documentation for IT Service Vision.

Note: The appropriate table to add the formula variable to is the table that contains the variables on which the formula variable is based. In addition to making the Kept status of the formula variable **Yes**, remember to check that the variables on which the formula variable is based also have Kept status set to **Yes**.

Adding a Formula Variable By Using the Interactive Interface

The formula variables UDAY and UWEEK can be useful when reporting on data in your PDB. To create them if they do not already exist:

1. Follow this path from the main menu:

Administration -> Manage Tables ->select the table -> click right mouse button -> Properties -> General -> Variables

2. For one formula variable, for instance UWEEK, select **Tools** and **Create Formula**. IT Service Vision displays the Create New Formula window. Type in the variable name UWEEK and select **OK**. IT Service Vision displays the Make/Edit Formula window. Select **General**.
3. Select the levels at which you want the formula calculated. In this case, select **Detail**, **Day**, **Week**, **Month**, and **Year** in the **Apply to level** field.

Note: If you choose to define a formula variable for multiple levels, you must be sure that any variables on which it is based have a Kept status of **Yes** and any statistics on which it is based are selected at those levels and their base variables have a Kept status of **Yes**.

4. Specify the length. In this case, type over the 8 with a 6 in the **Length** field.
5. Select the data type. In this case, select **Numeric** in the **Type** field.
6. Select the Kept status. In this case, select **Yes** in the **Kept in PDB** field.
7. Specify the label. In this case, type **Week** in the **Label** field.
8. Specify the description. In this case, in the **Description** field type **The Sunday immediately previous to the date in DATETIME**.
9. Select **Formula Source**. If you have questions about what constitutes a valid formula, select **Help**. You are presented with a largely blank window with numbers on the left side. Type in the formula to calculate values for the formula variable; in this case define UWEEK as follows:

```
UWEEK = datepart(datetime) - weekday(datepart(datetime)) + 1;
```

and then select **Check Code**. The results of the check display in the message area at the bottom of the SAS window.

10. Select **Advanced** and provide the subject and format, if applicable. In this case, select the right arrow that is associated with the Format field, **date7.**, and **OK**. Edit the format to `DATE .` and then select **OK**. Note that the new formula variable is now on the list of variables.

Repeat these steps to define another formula variable, UDAY, which has the formula:

```
UDAY = weekday(datepart(datetime));
```

and the description number of the day of the week (1-7), where 1=Sunday, and defaults for the remaining values.

To return to the main window, select **Close -> OK -> Close -> OK**.

Note: Your views will be rebuilt.

To examine the effect of defining the formula variable, follow this path from the main window:

Administration -> Examine PDB Data -> *select a table/level combination that you want to check* -> click right mouse button -> View Data

Use the scroll bars to view the UDAY and UWEEK values for each observation (if you have data).

Note: For other browsing methods, see the information on the Print Report Style in the GUI or see the information about the %CPPRINT macro in the Macro Reference documentation for IT Service Vision.

Note: Formula variables do not have selectable statistics. If you want statistics on a formula variable, you must define the statistics as formula variables at the desired non-detail levels.

Cabletron SPECTRUM Appendix 4: Defining IT Service Vision Tables for SPECTRUM Data

There are a number of methods for defining IT Service Vision tables for SPECTRUM data. For instance,

- IT Service Vision supplies table definitions for many common SPECTRUM model types. You can use those table definitions directly (by adding them from the master data dictionary to the PDB's data dictionary). Or you can use those table definitions as templates for a table definition for a different model type.
- If a supplied table definition is not available for a model type whose data you want to use,
 - i. you can create table definitions for the new model type by using the IT Service Vision server's interactive interface. After the table definition is created, you can use it as you would use any supplied table definition.
 - ii. you can use the IT Service Vision %CSATR2DD macro to create CPDDUTL control statements based on information that you export from SPECTRUM. When you run %CPDDUTL on the control statements, %CPDDUTL creates the corresponding table definition for that model type. After the table definition is created, you can use it as you would any supplied table definition.

This appendix describes a simplified version of the input, syntax, and parameters of the %CSATR2DD macro. If you need more information about the %CSATR2DD macro, see the Macro Reference documentation for IT Service Vision.

For information on %CPDDUTL, see the Macro Reference documentation for IT Service Vision.

Input Required to Run the %CSATR2DD Macro

To create CPDDUTL control statements, the %CSATR2DD macro requires the statistics data set (which you exported earlier in this chapter), the model data set (which you exported earlier in this chapter), the model type data set and the attributes data set (which you would need to export in the same way that you exported the other two data sets).

Using these data sets as input, %CSATR2DD writes CPDDUTL CREATE VARIABLE control statements for each variable in the statistics data set.

For more information on exporting data from SPECTRUM into SAS data sets, see [Section 1, Task 2: Start Your Data Collection Software](#).

For information on %CPDDUTL, see the Macro Reference documentation for IT Service Vision.

Syntax of the %CSATR2DD Macro

The %CSATR2DD macro has the following keyword syntax

```
%CSATR2DD(  MODTYP= ,      ATTR= ,      STAT= ,      MODEL= ,  
            EXTATTR= ,    OUTFILE= ,    FORMATS=NO ,  
            WHICH AID= ,   WHICHMTH= ,   WHICHTAB=NEW , WHICHVAR=ALL ,  
            BIGTEMP= ,    DATSTAMP=YES , MAXNAMES=512 ) ;
```

The important parameters are described below. For information about the others, see the Macro Reference documentation for IT Service Vision. For the definition of the `specgway` libref, see [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#).

■ MODTYP=

This parameter identifies the model type data set that you exported using SPECTRUM. Use a two-level name in the form *libref.datasetname*, where the SPECTRUM default *datasetname* is `modtyp` whether you did the export with SDE (SPECTRUM version 4.x) or Gateway (SPECTRUM version 3.x). For example:

```
MODTYP=specgway.modtyp
```

where `specgway` is the library name that you defined, before running this macro, by using the `libname` statement. `specgway` should point to the directory to which you exported SPECTRUM data.

Note: Each observation in the model type data set maps one model type handle to the corresponding model type name.

■ ATTR=

This parameter identifies the attributes data set that you exported using SPECTRUM. Use a two-level name in the form *libref.datasetname*, where the SPECTRUM default *datasetname* is `attr` if you did the export with SDE and `attrdesc` if you did the export with Gateway. For example:

```
ATTR=specgway.attr
```

Note: Each observation in the attr data set maps one attribute ID (in hex) to the corresponding attribute name.

■ STAT=

This parameter identifies the statistics data set that you exported using SPECTRUM. Use a two-level name in the form *libref.datasetname*, where the SPECTRUM default *datasetname* is `stat` whether you did the export with SDE or Gateway.

```
STAT=specgway.stat
```

Note: Each observation in the statistics data set contains the values of the attributes logged at a given time for a given model.

■ MODEL=

This parameter identifies the model data set that you exported using SPECTRUM. Use a two-level name in the form *libref.datasetname*, where the SPECTRUM default *datasetname* is `model` whether you did the export with SDE or Gateway. For example,

```
MODEL=specgway.model
```

Note: Each observation in the model data set maps one model handle (in hex) to the corresponding host name (which is the same as a value of the variable `MACHINE` in IT Service Vision).

■ OUTFILE=

This parameter is the name of the file containing the CPDDUTL control statements that are generated by %CSATR2DD. This file is a flat file suitable for use with the %CPDDUTL macro.

■ FORMATS=NO

Setting the value of this parameter to YES causes the macro to generate format definitions for user formats `uucal2n`, `uucmt2m`, and `uucmt2x`. These user formats are used only with previous releases of SAS/CPE; they are unnecessary with the current release.

■ WHICHID=

This parameter is used to subset the data so that the resulting table definitions reflect only the attributes that you select.

The value of this parameter is a list of the attribute ID values to be included in the table definition. For example:

```
WHICHAID=attrid in (65134 65980 64900)
```

selects only the variables with the indicated values of attrid.

■ WHICHMTH=

This parameter is used to subset the data so that the resulting table definitions reflect only the model types that you select.

The value of this parameter is a list of the model types that should be used for creating tables. For example:

```
WHICHMTH=mth in (65134 65132)
```

selects only those device types for table creation. Records for models of other types will not be used to determine which attributes are included in the table.

■ WHICHTAB=NEW

This parameter is used to subset the data so that the resulting definitions reflect only the tables that are not already known to IT Service Vision. Setting this parameter to ALL will cause the macro to create CPDDUTL control statements for all the different tables (model types) encountered in the data sets.

Leaving the value of this parameter equal to NEW will cause the macro to generate CPDDUTL statements only for model types that do not already have matching table definitions in IT Service Vision.

■ WHICHVAR=ALL

This parameter is used to subset the data so that the resulting table definitions reflect only the variables that you select.

Leaving the value of this parameter equal to ALL will cause the macro to generate CPDDUTL control statements for all attributes encountered in the data sets.

Setting this parameter to NEW will cause the macro to generate CPDDUTL control statements only for attributes that were not previously defined in IT Service Vision.

For example, this program (which you can type in and submit from the SAS PROGRAM EDITOR window) generates an IT Service Vision table definition that contains all the attributes exported into `specgway.stat` and lists the table definition in the SAS LOG window:

```
* Define the specgway libref;
  libname specgway '/usr/spectrum/export.output' ;

* Generate table definition statements;
%csatr2dd(modtyp=specgway.modtyp,
  attr=specgway.attr,
  model=specgway.model,
  stat=specgway.stat,
  outfile=/tmp/table.ddutl,
  whichtab=ALL,
  whichvar=ALL) ;

* Read table definition statements into %CPDDUTL;
%cpddutl (filename=/tmp/table.ddutl, list=y) ;
```

If you want to write the SAS log to an external file, see [Prerequisite 1: You have a printed copy of the table definition\(s\) in Section 2, Task 1: Customize and Verify your Test PDB](#) for the steps.

Cabletron SPECTRUM Appendix 5: Editing Model Types for SPECTRUM

1. Stop SpectroSERVER.

To edit models in SPECTRUM, you will need to stop SpectroSERVER.

Using the SPECTRUM Control Panel, stop the SpectroSERVER by selecting **Stop SERVER**. Select **OK** to confirm that you want to shut the server down. The server is stopped when messages similar to the following appear in the message window of the Control Panel

```
Closing /usr/spectrum/SS/SpectroSERVER database...  
/usr/spectrum/SS/SpectroSERVER database has successfully shut down.
```

and the Status field is set to Inactive.

2. Check the variables in the IT Service Vision table for that model type.

To determine which attributes have been defined for a particular model type (in the IT Service Vision table that corresponds to that model type), from the IT Service Vision main window follow this path:

Administration -> Explore Tables/Variables -> Collector: Selected -> SPECTRUM -> OK -> *select the table of interest* -> Variables -> OK

Follow this path

select the variable of interest -> click right mouse button -> View Variable Properties

Check that the Kept status is Yes for any variable whose data you want to analyze. If it is not, change it to Yes. If there are attributes that you want to keep in your PDB that do not appear in the IT Service Vision supplied table definition, you will need to add a variable to your PDB for that attribute.

3. Start the SPECTRUM Model Type editor.

After you stop the server, from the SPECTRUM Control Panel start the Model Type Editor by selecting

Configure -> Model Type Editor

4. Select the model type to be changed.

To select the model type to be changed, in the editor select

File -> File Model Type...

SPECTRUM displays the Find Model Type window.

Type the model type name (or a string that contains part of the name of the model type) in the Name or Handle field and then select **Apply**. SPECTRUM displays a list of all model types that contain a match to the string that you entered. Select the correct model type name from the list and then select **OK**.

5. List the attributes for this model type.

Follow this path:

File -> Examine Attributes

SPECTRUM displays the Attribute View window. The window lists all available attributes for this model type.

6. Select an attribute whose logging status you want to change, and change it.

- a. To change the logging flag for a given attribute, select the attribute name from the list. The attribute selected should be one of the variables in the corresponding IT Service Vision table. For example, select avgBusy1.
- b. To change the attribute's flag, select

File -> Open Attribute...

SPECTRUM displays the Attribute Edit View window.

- c. To enable logging, depress the Logged radio button in the Extended Flags section of the window.
- d. To save the change, select:

File -> Save & Close Attr Edit.

SPECTRUM displays a warning similar to the following:

```
You are not the owner of this attribute.  
You will not be able to export this model type and this attribute  
later.  
Are you sure you want to save these changes?
```

This message indicates that the changes to the model type will not be saved if SPECTRUM is upgraded. The changes are only valid for this instance of SPECTRUM and its associated database. Therefore, remember which model types and attributes that you changed so that you can make these changes again if SPECTRUM updates the model type definitions.

Repeat for each attribute that you want to use from the IT Service Vision table and is not enabled for logging in SPECTRUM.

7. Save all changes.

When you are finished, select

File -> Save All Changes

The editor returns you to the Model Type View window.

8. Exit from the editor.

Select

File -> Save to Permanent Catalog -> File -> Exit MTE

9. Restart SpectroSERVER.

To restart the SpectroSERVER, select **Start SERVER** in the SPECTRUM Control Panel. Check to see that these new attributes are being logged for a model of this particular model type.

Cabletron SPECTRUM Appendix 6: Mapping SPECTRUM Data Types to an IT Service Vision PDB

To read SPECTRUM data into an IT Service Vision PDB, it is necessary to tell SPECTRUM to write out the same data that you tell the IT Service Vision server to read in. If what is written does not correspond to what is read, the result is generally that %CSPROCES reads no data (or a smaller amount of data than you expected) into your PDB.

In the information that the %CSPROCES macro writes to the SAS log, there are messages indicating the number of observations read at various stages by the %CSPROCES macro and, at the end, a report on the number of observations kept and the date range of the data.

The following is a list of problems that may be indicated in the log, with an explanation of what they mean (and what to do about them):

- **The message "NOTE: CSIIPDI not initialized" appears.**

This message means that some variables that are defined in the IT Service Vision table are not being logged and exported (as attributes) by SPECTRUM.

You will need to edit the SPECTRUM model type to enable logging of these attributes and re-export the data. To edit the SPECTRUM model type, see [Cabletron SPECTRUM Appendix 5: Editing Model Types for SPECTRUM](#).

- **The message "ERROR: Negative value for counter CSIIPFD=-873173447 is invalid" appears.**

Messages like this refer to data in the form of counters (that is, monotonically increasing values). Certain releases of SPECTRUM log negative numbers when the values in the counters become very large. See `!sasroot/misc/cpe/cscsivew` for a workaround until Cabletron fixes this problem.

- **The date range of the data that were read is not correct.**

At the end of the messages that %CSPROCES writes to the SAS log, there is a report on the number of observations read in this %CSPROCES run and the date range of the data that %CSPROCES kept. If the date range is not what you expect, you may have exported the wrong range.

You can check the date range in the input (`specgway.stat`) data set by typing this command on the PROGRAM EDITOR window's command line after %CSPROCES completes:

```
fsview specgway.stat
```

and pressing ENTER or RETURN. Then, at the FSVIEW window's command line, type (do not forget the dot "." after datetime):

```
format ts "datetime."
```

and press ENTER or RETURN.

The variable named TS contains the date/time stamp of each observation. These date/time stamps should correspond to the date range that you specified when you exported the data. See [Section 1, Task 1: Start the IT Service Vision Server Software](#), Action 2.b.viii in this chapter to check the Export Range on the Export Definition Description that you used to export the data.

- **The data were read but no observations were kept.**

If the SAS log shows that observations were read by %CSPROCES, for example:

```
NOTE: Processing RECNUM=1000 MACHINE=Rtr1 DATETIME=15JUN96:01:09:02
```

but the report at the end of the log shows that no observations were kept, for example:

NOTE:	+-----						
NOTE:	Table	AgeLim	ObsIn	ObsKept	RBDF	LastIn	First-LastKept
NOTE:	+-----						
NOTE:	CCIRTR	10	0	0	N/A	N/A	N/A
NOTE:	+-----						

then

- the model types that you exported do not match the IT Service Vision tables that you named on the %CSPROCES macro (or selected, if you defined the process step through the server's interactive interface), or
- the attributes that you exported do not match the variables defined in the IT Service Vision tables that you named on the %CSPROCES macro (or selected, if you defined the process step through the server's interactive interface).

Each of these scenarios is described in some detail below.

Note: Earlier in this appendix we dealt with the case of not exporting variables for the tables that you named on the %CSPROCES macro. This case is the reverse of that: not defining the correct variable for an exported attribute.

- **Investigate whether the IT Service Vision table that you are using is the correct one for the SPECTRUM model type that you are using.**

A SPECTRUM model type is mapped to an IT Service Vision table by matching the SPECTRUM model type handle and the IT Service Vision Table ID Number (TABIDNUM). The following steps walk you through checking these values and making the required corrections if they do not match:

1. Creating a few SAS formats that map model handles and model type handles to each other and to their names will help you make sense of the `specgway.stat` data set.

In the SAS PROGRAM EDITOR window, type the following starting at the first line in the body of the window:

```
/* This assumes that you have exported the model type, model, and
   statistics data sets and that they are in the directory that th
   specgway libref points to, and that they are named
   specgway.modtyp, specgway.model, and specgway.stat . */

libname specgway '/usr/spectrum/export.output' ;

filename temp temp;

/* Map SPECTRUM model type handle to model type name */
data _null_;
  set specgway.modtyp end=lastrec;
  file temp; if _n_=1 then put 'proc format;
  value mth2mtn';
  put mth '=' mtname ' ';
  if lastrec then put '; run;';
run;
%include temp;

/* Map SPECTRUM model handle to model type name */
data _null_;
  set specgway.model end=lastrec;
  file temp;
  if _n_=1 then put 'proc format; value mh2mtn';
  put mh '=' mth mth2mtn. ' ';
  if lastrec then put '; run;';
run;
%include temp;
```

```

/* Map SPECTRUM model handle to model type handle */
data _null_;
  set specgway.model end=lastrec;
  file temp;
  if _n_=1 then put 'proc format; value mh2mth';
  put mh '=' mth;
  if lastrec then put '; run;';
run;
%include temp;

```

and submit the program (by typing SUBMIT on the command line and pressing ENTER or RETURN) after the %CSPROCES macro completes.

- a. When the program completes (as indicated in the SAS log), issue this command

```
fsvview specgway.stat
```

- b. Type bye.

To include the file in the PROGRAM EDITOR window, issue this command:

```
spectrum.gs
```

2. If the FSVIEW window has pull-down menus, select **Globals**, **Options**, and **Command line** to get a command line. To see the model names that are in the data, at the FSVIEW window's command line prompt type this:

```
format mh 'cscmo2x.'
```

and press ENTER or RETURN. The variable named MH contains the model handle. These values are now displayed as a names rather than numbers. Scroll down through the data observing the names now show in the model handle field. These should be the models you exported. If they are not, check your Export Definition Description and your model definition in SPECTRUM.

3. To display the names of the model types to which these models belong, at the FSVIEW window's command line prompt type this:

```
format mh 'mh2mtn.'
```

and press ENTER or RETURN. Model type names now display for the values of MH.

These should match exactly the external names of the tables in your PDB. To check the external names, follow this path from the IT Service Vision main window:

Administration -> Manage Tables -> select the table -> click right mouse button -> Properties -> Advanced

The IT Service Vision table's external name is in the `External Name` field. These table names should be the ones specified in the %CSPROCES macro or in the GUI when selecting **Administration -> Process Data - Wizard**.

4. If the model type name matches the table's external name exactly (case sensitive), but data did not read into the table during the %CSPROCES run, you need to compare the model type handle of the models in the `specgway.stat` data set to the IT Service Vision table IDs.

To see the model type handles (instead of the model type names) of the data in the `specgway.stat` data set, at the FSVIEW window's command line prompt type this:

```
format mh 'mh2mth.'
```

and press ENTER or RETURN. The values for MH now display as model type handles.

Compare this number to the ID number of the corresponding IT Service Vision table. To check table ID numbers, follow this path from the IT Service Vision main window:

Administration -> Manage Tables -> *select a table* -> click right mouse button -> Properties

and look in the ID Number field, which is approximately in the middle of the window.

The table's ID Number needs to match the model type handle in the `specgway.stat` data set. If it does not match, you need to replace this IT Service Vision table with the table that does have a matching table ID number. To search all IT Service Vision supplied tables for a table with this table ID Number (model type handle), follow this path from the IT Service Vision main window:

Administration -> Explore Tables/Variables

In the Search field, type `TABIDNUM=n`, where n is the model type handle. Then select **OK**. For example, this expression

```
TABIDNUM = 124556
```

lists the table with `TABIDNUM=124556`.

To add this table to your PDB, follow this path:

**Administration -> Manage Tables -> File -> Add Table Definition -> Collector:
Specified -> SPECTRUM -> OK -> *scroll to the table name and select it* -> click right
mouse button -> Add Table to Active PDB**

When the Migration in Progress window closes, select **Close**. The Manage Tables window displays. Notice that the new table is now added to the list of tables in the active PDB.

If you found no supplied table for a given table ID number (model type handle), you need to create a user-defined table. See [Cabletron SPECTRUM Appendix 4: Defining IT Service Vision Tables for SPECTRUM Data](#) in this chapter for information on using the `%CSATR2DD` macro to generate the appropriate CPDDUTL control statements for you from the information in the `specgway.stat` data set.

■ Investigate whether the IT Service Vision variables that you are using are the correct ones for the SPECTRUM attributes that you are using.

If the IT Service Vision tables that you listed on your `%CSPROCES` macro match the SPECTRUM model type handles of the models that you exported, but still no observations are kept at the end of the `%CSPROCES` run, in your IT Service Vision table you may not have variables defined for the attributes you exported.

A SPECTRUM attribute is mapped to an IT Service Vision variable by matching the SPECTRUM attribute ID number and the IT Service Vision Variable ID Number (VARIDNUM). The following steps walk you through checking these values and making the required corrections if they do not match:

1. You should check the attribute ID numbers that you are exporting in two different places within SPECTRUM: first in the SPECTRUM internal list of attributes to be exported, and then again in the actual exported statistics data set.
2. To display the SPECTRUM list of attributes to be exported for a particular model, from the Topology View window follow this path

***select the model* -> View -> Icon Subviews -> Model Information**

The list of attributes under "LOGGED" shows the attribute name and attribute ID number of the attributes

that are to be exported; for example:

```
TCP_Retrans_Min_Val 1007f
```

These attribute ID numbers should match the attribute ID numbers in the actual exported statistics data set and in your PDB's data dictionary. Keep this window up so that you can compare attribute ID numbers from these other sources in the next steps.

3. To display the attribute ID numbers that are in the exported statistics data set, allocate the statistics data set library as you did in [Section 1, Task 3: Create a Test PDB and Process, Reduce and Report on Your Data](#). Then, make the PROGRAM EDITOR window the active window and issue the FSVIEW SPECGWAY . STAT command.

The window displayed shows the data in the statistics data set. The column labeled TS is TimeStamp; the column labeled MH is ModelHandle. The other columns contain the values of the model's attributes at the indicated time. The hexadecimal representation of the attribute ID number (with the first hex digit translated to a letter or underscore) is used as the label (variable name) for the column. The first character of the variable name is translated as follows:

0->_	4->D	8->H	C->L
1->A	5->E	9->I	D->M
2->B	6->F	A->J	E->N
3->C	7->G	B->K	F->O

Thus, values for attribute ID number 65,663 (which is 0x1007f in hex) would appear in a column labeled _001007F.

These attribute ID numbers should match the attribute ID numbers in in your PDB's data dictionary. Keep this window up and go on to the next step to display the variable ID numbers (the IT Service Vision version of the attribute ID numbers) in your PDB's data dictionary.

4. To display the variable ID numbers for the corresponding IT Service Vision table, follow this path from the IT Service Vision main window:

Administration -> Manage Tables -> select the table -> click right mouse button -> Properties -> select a variable -> click right mouse button -> Properties -> Advanced

5. The IT Service Vision variable's ID Number (which is in decimal in the Varidnum field) needs to match the SPECTRUM attribute ID number (which is in hex). If it does not, you need to add the correct variable definition to the IT Service Vision table. If this attribute has already been defined in another IT Service Vision table, you can copy the variable definition.

To search all IT Service Vision supplied variables for one with this attribute ID number, follow this path from the IT Service Vision main menu

Administration -> Explore Tables/Variables -> Collector: All -> Variables

Edit the Search field Clause" expression to select the variable ID number in which you are interested and then select **OK**. For example, this expression:

```
tablenm="C" and varidnum = 01007fx
```

lists all variables with VARIDNUM (attribute ID number) of 0x01007f in all tables whose names start with "C" (which all supplied SPECTRUM tables do).

- If there is already at least one variable already defined in IT Service Vision for the attribute of interest, you can copy the variable by following this path from the IT Service Vision main menu

Administration -> Manage Tables -> select the table to which you want to add the variable -> click right mouse button -> List Variables -> Tools -> Copy

Variable -> Collector: Selected -> SPECTRUM -> OK -> OK -> *scroll to the variable and select it* -> *click right mouse button* -> Copy Variable to Current Table -> Close

IT Service Vision re-displays the list of variables in the table, and the new variable is now in the list. Select **Close -> Close -> OK** to return to the main window.

Alternatively, you can add the variable to the table by using %CPDDUTL control statements. For more information on running the %CPDDUTL macro, see the Macro Reference documentation for IT Service Vision. Use the form of the CREATE VARIABLE control statement that includes the LIKE= parameter.

- If you found no supplied variable for a given variable ID number (attribute ID number), you need to create a user-defined variable. See [Cabletron SPECTRUM Appendix 4: Defining IT Service Vision Tables for SPECTRUM Data](#) for information on using the %CSATR2DD macro to generate the appropriate CPDDUTL control statements for you.

OpenView and NetView Appendix 1: Recommended Metrics

Note: The tables whose names start with HN2 in IT Service Vision contain MIB-II data collected by HPOV/NVAIX. This document describes working with tables HN2IFT and HN2NIX.

MIB-II Table/Group Name	Which Metrics	PDB Table Name
mib-II.egp	All metrics (1.3.6.1.2.1.8.*)	HN2EGP
mib-II.icmp	All metrics (1.3.6.1.2.1.5.*)	HN2ICM
mib-II.interfaces	All metrics (1.3.6.1.2.1.2.1)	HN2INF
mib-II.ip	All metrics HN2IP_ (1.3.6.1.2.1.4.*) except ipDefaultTTL and ipReasmTimeout	HN2IP_
mib-II.non-indexed	See HN2EGP, HN2ICM, HN2INF, HN2IP, HN2SNP, HN2SYS, HN2TCP, and HN2UDP	HN2NIX
mib-II.snmpstat	All metrics (1.3.6.1.2.1.11.*)	HN2SNP
mib-II.system	All metrics HN2SYS (.3.6.1.2.1.1.*)	HN2SYS
mib-II.tcp	All metrics (1.3.6.1.2.1.6.*) except tcpActiveOpens, tcpMaxConn, tcpAttemptFails, tcpEstabResets, tcpPassiveOpens, tcpRtoMax, and tcpRtoMin	HN2TCP
mib-II.udp	All metrics (1.3.6.1.2.1.7.*)	HN2UDP

OpenView and NetView Appendix 2: Alternate Data-Logging Method

This is an alternate version of the task in [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#). In this version of the task, you create your own `snmpCol.conf` file (the file of variables for which data are to be collected and the nodes on which the data are to be collected). You also manually stop and start the `snmpCollect` daemon.

Note: The procedures and the examples presented here are provided as a convenience to IT Service Vision customers. The methods outlined here are neither documented, suggested, nor supported by any Hewlett Packard documentation or IBM documentation. Although these procedures and examples have been tested with HP Open View Release 3.0 and NetView for AIX Release 4.0 and are used at SAS Institute, the Institute offers no official support for them.

Prerequisites

1. **SAS is installed.**

See [Section 1, Task 1: Start the IT Service Vision Server Software](#).

2. **The IT Service Vision server is installed.**

See [Section 1, Task 1: Start the IT Service Vision Server Software](#).

3. **You know which nodes support at least Level 3 management and MIB-II.**

See [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#).

Actions

1. **Make copies of three files from the `!sasroot/misc/cpe` directory.**

For illustration purposes only, the examples in this action assume that `!sasroot` is in `/usr/local/sas` and you are using `/tmp` as a work area.

Execute the following commands:

```
cd /usr/local/sas/misc/cpe
cp csgmkcnf /tmp/csgmkcnf
cp csghosts /tmp/csghosts
cp csgvars /tmp/csgvars
```

Note: The `csgmkcnf` file contains a shell script that builds the `snmpCol.conf` file. The `csghosts` file will contain a list of host names of all the hosts from which you want to collect data. The `csgvars` file contains the recommended MIB metrics/variables in the `snmpCol.conf` format.

2. **Make the `csgmkcnf` file executable.**

Issue this command:

```
chmod 755 /tmp/csgmkcnf
```

3. **Edit the `csghosts` file so that it contains the host names of all the hosts from which you want to collect data.**

Each fully qualified host name must be listed on a separate line, starting in column 1. For example, the lines for nodes `dimes`, `swimmer`, and `winter` might look like this:

```
dimes.unx.sas.com
swimmer.unx.sas.com
winter.unx.sas.com
```

4. Create an `snmpCol.conf` file.

Issue this command:

```
/tmp/csgmkcnf
```

5. Back up the existing `/usr/OV/conf/snmpCol.conf` file.

The existing `snmpCol.conf` file contains your current collection configuration. (If you made any modifications in the configuration, see [Section 1, Task 3: Create a Test PDB and Process, Reduce, and Report on Your Data](#), the changes that you made are included in this file. In this step you will save those changes but not use them.)

Issue this command:

```
cd /usr/OV/conf
cp -p snmpCol.conf snmpCol.conf.orig
```

6. During the following actions, monitor the contents of the `snmpCollect` trace file.

Messages are appended to the `snmpCollect` trace file as data collection takes place. The best way to monitor these messages is with the following steps:

- a. Open a new xterm window.
- b.
- c. Issue these commands in the new window:

```
cd /usr/OV/log
tail -f snmpCol.trace
```

Messages will be displayed as they are appended to the trace file.

7. Begin logging data using the new `snmpCol.conf` file.

- a.
- b. To stop the `snmpCollect` daemon, issue this command:

```
/usr/OV/bin/ovstop snmpCollect
```

- c. To copy your config file over the existing config file, issue this command:

```
cp /tmp/snmpCol.conf /usr/OV/conf/snmpCol.conf
```

- d. To restart the `snmpCollect` daemon, issue this command:

```
/usr/OV/bin/ovstart snmpCollect
```

Note: If at any time you want to return to using your former configuration, repeat this action except issue this command as the copy command:

```
cp -p /usr/OV/conf/snmpCol.conf.orig /usr/OV/conf/snmpCol.conf
```

8. Watch the trace file as `snmpCollect` runs.

- You may see a message that looks like this:

```
Wed Aug 16 11:30:19 1995 : Next object after ipRoutingDiscards
(.1.3.6.1.2.1.4.23) is not supported by agent on dimes.unx.sas.com.
Use snmpwalk or MIB browser to verify the object exists on this node
```

In this example, the message indicates that `snmpCollect` will not be able to collect `ipRoutingDiscards`

on `dimes.unx.sas.com` because `dimes` is not a router. The `snmpCollect` daemon ignores this metric for `dimes` and proceeds normally with collecting the rest of the metrics for `dimes`.

- Another kind of message you may see is like this:

```
Tue Jun 06 11:41:40 1995 : swimmer.unx.sas.com doesn't reply to a 100 object
but responds to sysUpTime. Be sure SNMP timeouts are not
set too small (SNMP interval: 0.80s retry: 3).
```

In this example, the message indicates that the `swimmer` host could not respond within the timeout period.

The `snmpCollect` daemon bundles multiple requests in a single Protocol Data Unit (PDU). Depending on your network, the agent on the host may not be able to respond within the timeout and retry parameters. There are two ways around this problem:

- The better fix is to reduce the number of requests that `snmpCollect` makes per PDU. The file `/usr/OV/lrf/snmpCol.lrf` specifies the number of requests that `snmpCollect` makes per PDU. The following steps describe how to edit the file and then replace the existing file with the edited file:

- i. Make a backup copy of `/usr/OV/lrf/snmpCol.lrf` by issuing this command:

```
cp -p /usr/OV/lrf/snmpCol.lrf /usr/OV/lrf/snmpCol.lrf.orig
```

- ii. Change the number of objects per PDU by editing the file `/usr/OV/lrf/snmpCol.lrf`. Specify the number *nn* of objects by adding

```
-m nn
```

to the third field on the second line of `snmpCol.lrf`. The value of *nn* to use depends on your network load. A larger number means `snmpCollect` can "batch" more requests and improve performance of `snmp get` requests.

The updated file should look something like this:

```
snmpCollect:/usr/OV/bin/snmpCollect:
OVs_YES_START:trapd,ovwdb,ovtopmd:-m 20:OVs_WELL_BEHAVED:120:
```

- iii. Stop the `snmpCollect` daemon with this command:

```
/usr/OV/bin/ovstop snmpCollect
```

- iv. Re-register `snmpCollect` with this command:

```
/usr/OV/bin/ovaddobj /usr/OV/lrf/snmpCol.lrf
```

- v. Re-start the `snmpCollect` daemon with this command:

```
/usr/OV/bin/ovstart snmpCollect
```

- vi. Verify that `snmpCollect` was properly started:

```
ps -ef | grep snmpCollect
```

You should see an entry containing, for this example,

```
snmpCollect -m 20
```

- vii. Check the trace file, in the same way as described earlier in this action, for messages.

- Another way to fix the timeout problem is to increase the global default timeout by following this path

from the main menu in OVW:

Options -> SNMP Configuration -> Global default -> *change the value in the timeout*
-> Replace -> OK

OpenView and NetView Appendix 3: OVW-Specific Tips for Reporting

Subsetting by Using a Format in Local or Global Where

If you have too many values of interest to fit in a WHERE expression or if a variable needs some sort of transformation (for example, a table lookup) before you can construct a WHERE expression, you may want to use a user-written format.

In a simple WHERE expression (see [Subsetting by a Simple Local Query Expression](#) in Open Systems Appendix 2: Tips for Reports or see [Subsetting by a Simple Global Query Expression](#) in Open Systems Appendix 2: Tips for Reports), you may want to construct an expression using the contains operator. For instance,

```
variable-name ? 'value'
```

where ? means contains. You can use the contains operator only if *variable-name* is of IT Service Vision interpretation type STRING.

However, with a format some IT Service Vision numeric variables can be transformed into more meaningful string values, and you can subset based on the transformed values.

For example, you can subset on the machines that provide a specific service, such as IP gateway service. In HP-OV table HN2NIX, the service is described in the SYSSERV variable. Because the SYSSERV variable is numeric, you cannot use a WHERE expression like this:

```
SYSSERV ? 'IPGATEWAY'
```

However, if you convert the value of the SYSSERV variable to a string by using the SYSSERV format supplied in IT Service Vision, you can test on the converted value of SYSSERV. In that case, you can use a WHERE expression like this:

```
put(sysserv,sysserv.) ? 'IPGATEWAY'
```

where the put function causes the original value of the SYSSERV variable to be replaced by the value produced by the application of the SYSSERV format.

Finding an IT Service Vision Supplied Format

To find the IT Service Vision supplied format, if any, for converting a numeric metric to a character string, follow this path from the IT Service Vision main window:

Administration -> Manage Tables ->
select your table -> click right mouse button ->
Properties -> Variables ->
select your variable -> click right mouse button-> Properties -> Advanced

where *the variable* is the variable for which you want to find a format and *the table* is the table in which that variable resides. The Format field displays the supplied format, if any.

Creating a User-Written Format

Explanations that are associated with supplied report definitions sometimes refer you to IT Service Vision supplied example programs for building user-written formats. You can submit the example programs as-is in the PROGRAM EDITOR window, or modify them, or write format-generating programs from scratch.

The csgfmt2 example program is an example of a user-written format that provides the same capability of testing for a string, but the user format is based on the value of MACHINE rather than SYSSERV (and thus can be used on tables that do not contain the SYSSERV variable). The csgfmt2 program generates a format named \$USVCS, which translates the MACHINE value to a string. This must be a user-written format because it uses the values of MACHINE *at your site* as read from your DETAIL.HN2NIX view.

To create the \$USVCS format:

1. Make the SAS PROGRAM EDITOR window the active window.
2. Issue this command

```
include !sasroot/misc/cpe/csgfmt2
```

SAS displays the program in the body of the PROGRAM EDITOR window.

The program has a comment that describes the PDB table and level on which it runs to create the format. In this case, the program is designed to run on SNMP MIB-II data in the DETAIL.HN2NIX table.

3. To modify the program so that the format is saved in SITELIB.CPFMTS and, thus, is available to any IT Service Vision user at your site, in the program portion of csgfmt2, edit this line:

```
if _n_=1 then put "PROC FORMAT;" /
```

to be

```
if _n_=1 then put "PROC FORMAT library=sitelib.cpfmts;" /
```

Note: If you do not save the format in SITELIB, the format lasts only as long as your current SAS session. To be able to save the format in SITELIB, prior to submitting this program you must invoke the IT Service Vision server with SITEACC=WRITE.

Note: You may want to keep the FORMAT procedure code as part of your daily job so that it is easy to remember how to update the \$USVCS format.

4. To build the format, select **Locals -> Submit**.
5. To use this format, in the Local Where expression or Global Where expression (see above), you can construct a WHERE expression like this:

```
put(machine,$USVCS.) ? 'IPGATEWAY'
```

Note: You may find it helpful to look at other IT Service Vision supplied examples for building formats in other ways. For instance, the csgfmt1 example program builds a format \$UMACHTY that contains a user-supplied list of key machines. To use this format, you can construct a WHERE expression like this:

```
put(machine,$UMACHTY.) = 'SERVER';
```

Because this format uses a user-supplied list rather than an automatic list, you need to edit the program and rebuild the format if the list of key machines in your network changes.

Note: You may want to keep the FORMAT procedure code as part of your daily job so that it is easy to remember how to update the \$USVCS format.

Note: For more information on the example programs and their formats, see the comments in the programs. For more information on user-written formats in general, see "The FORMAT Procedure" in the *SAS Procedures Guide* in the documentation for your current version of SAS.

Using One or More Formula Variables

Formula variables are variables whose values are calculated (from the values of other variables) when you access the data. The values of the formula variables do not occupy disk space in the PDB.

You can add formula variables to PDB tables by using the the IT Service Vision server's interactive interface or by using the IT Service Vision %CPDDUTL macro. It is generally preferable to use the %CPDDUTL macro for your production PDB because the macro documents what you did exactly and enables you to add the variables to many tables easily and accurately.

For information on adding a formula variable using %CPDDUTL, see the Macro Reference documentation for IT Service

Vision.

Note: The appropriate table to add the formula variable to is the table that contains the variables on which the formula variable is based. In addition to making the Kept status of the formula variable Yes, remember to check that the variables on which the formula variable is based also have Kept status set to Yes.

Adding a Formula Variable By Using the %CPDDUTL Macro

IT Service Vision supplies an example program for using the %CPDDUTL macro to define formula variables. To view and optionally run it, follow these steps when IT Service Vision is running:

1. Make the PROGRAM EDITOR window the active window.
2. Issue this command:

```
include !sasroot/misc/cpe/csgfvar1
```

SAS displays the program in the body of the PROGRAM EDITOR window.

The program has a comment that describes the PDB table and level on which the program runs to create the formula. In this case, the program is designed to add formula variables to the HP-OV MIB-II tables HN2IFT and HN2NIX. UWEEK (a name chosen to be unique in this table and starting with U for user-defined formula) is defined to be the date of the Sunday preceding the DATETIME value. UDAY (a name similarly chosen) is defined to be the numeric day of the week (where 1=Sunday).

You can edit those formulas if you want to construct the values in a different way. For more about the construction of formula variables, follow this path from the main menu:

Administration -> Manage Tables -> select any table -> click right mouse button -> List Variables -> Tools -> Create Formula -> type TEST -> OK

IT Service Vision displays the Make/Edit Formula window. Select Help for information about formula variables and the source statements that define their formulas.

Note: You may find UWEEK and UDAY useful in other PDB tables; the other formula variables are rather specific to the specified tables.

Note: Formula variables do not have selectable statistics. If you want statistics on a formula variable, you must define the statistics as formula variables at the desired non-detail levels.

3. Submit the program by selecting **Locals -> Submit**.
4. To examine the effect of defining the formula variable, follow this path from the main menu:

Administration -> Examine PDB Data -> Levels: All -> select a table/level combination that you want to check -> click right mouse button -> View Data

You can use the scroll bars the UDAY and UWEEK values for each observation (if you have data).

Note: You can also use the Report Style PRINT in the GUI to view the data.

Note: For more information about the example programs, see the comments in the programs. For more information about the %CPDDUTL macro, see the Macro Reference documentation for IT Service Vision.

Adding a Formula Variable By Using the Interactive Interface

The formula variables UDAY and UWEEK can be useful when reporting on data in your PDB. To create them if they do not already exist:

1. Follow this path from the main menu:

Administration -> Manage Tables -> select the table -> click right mouse button -> Properties -> General -> Variables

2. For one formula variable, for instance UWEEK, select **Tools** and **Create Formula**. IT Service Vision displays the Create New Formula window. Type in the variable name UWEEK and select **OK**. IT Service Vision displays the Make/Edit Formula window. Select **General**.
3. Select the levels at which you want the formula calculated. In this case, select Detail, Day, Week, Month, and Year in the Apply to level field.

Note: If you choose to define a formula variable for multiple levels, you must be sure that any variables on which it is based have a Kept status of Yes and any statistics on which it is based are selected at those levels and their base variables have a Kept status of Yes.

4. Specify the length. In this case, type over the 8 with a 6 in the Length field.
5. Select the data type. In this case, select Numeric in the Type field.
6. Select the Kept status. In this case, select Yes in the Kept in PDB field.
7. Specify the label. In this case, type Week in the Label field.
8. Specify the description. In this case, in the Description field type The Sunday immediately previous to the date in DATETIME.
9. Select **Formula Source**. If you have questions about what constitutes a valid formula, select **Help**. You are presented with a largely blank window with numbers on the left side. Type in the formula to calculate values for the formula variable; in this case define UWEEK as follows:

```
UWEEK = datepart(datetime) - weekday(datepart(datetime)) + 1;
```

and then select **Check Code**. The results of the check display in the message area at the bottom of the SAS window.

10. Select **Advanced** and provide the subject and format, if applicable. In this case, select the right arrow that is associated with the Format field, **date7.**, and **OK**. Edit the format to DATE. and then select **OK**. Note that the new formula variable is now on the list of variables.

Repeat these steps to define another formula variable, UDAY, which has the formula:

```
UDAY = weekday(datepart(datetime));
```

and the description Number of the day of the week (1-7), where 1=Sunday, and defaults for the remaining values.

To return to the main window, select **Close -> OK -> Close -> OK**.

Note: Your views will be rebuilt.

To examine the effect of defining the formula variable, follow this path from the main window:

Administration -> Examine PDB Data -> select a table/level combination that you want to check -> click right mouse button -> View Data

Use the scroll bars to view the UDAY and UWEEK values for each observation (if you have data).

Note: For other browsing methods, see the information on the Print Report Style in the GUI or see the information about the %CPPRINT macro in the Macro Reference documentation for IT Service Vision.

Note: Formula variables do not have selectable statistics. If you want statistics on a formula variable, you must define the statistics as formula variables at the desired non-detail levels.

SunNet Manager and Enterprise Manager Appendix 1: Mapping SunNet Manager Metrics to IT Service Vision Tables and Variables

SunNet Manager logged data is read into IT Service Vision tables based on the name of the table. Select tables for your PDB that correspond to the data in your Sun Net Manager log files by matching the IT Service Vision table external name (determined via the IT Service Vision graphical user interface **Administration** -> **Explore Tables/Variables**) with the SunNet Manager table name (determined from Sun documentation, from the SunNet Manager interactive user interface, or from the SunNet Manager log file itself). For SunNet Manager 1.0 files, the external name appears as the second field in each log record. For example, log record:

```
D iostat.disk.694816221 192.100.6.151 100106 192.100.6.151 6091 695326091
900 0 0 mercury disk id000 . . .
```

is a record for the table with external name iostat.disk.

For SunNet Manager 2.0 and later, the second field in each logged record is blank, so the table name must be determined from the rpc number (field 4) and the table group name (field 14). The rpc number maps to the first name of the table (the part before the dot) using the /etc/rpc file on the workstation running SunNet Manager. The group name is the second name of the table (the part after the dot). For example, the 100107 in this log record:

```
D " " 192.9.214.1 100107 192.9.214.1 716827965 720001 716859485 716859486
21589875 0 64 bb1403 data " " . . .
```

would indicate that this record is for the table hostperf.data if /etc/rpc file on the workstation running SunNet Manager contained the following line:

```
hostperf 100107 na.hostperf
```

SunNet Manager and Enterprise Manager Appendix 2: Defining Tables from Your Own Schema Files

To read data logged by SunNet Manager using a schema file not supported by IT Service Vision, you must first create IT Service Vision dictionary entries for the tables of metrics defined in the schema file using the %CSSNMSCH and %CPDDUTL macros. Run these macros iteratively to create suitable definitions in your PDB for the tables in the schema file.

See the IT Service Vision Macro Reference documentation for instructions on using %CPDDUTL.

%CSSNMSCH Syntax

The %CSSNMSCH macro takes the following parameters:

```
%CSSNMSCH( schema=,  
            rpcids=,  
            snmnams=,  
            outfile= );
```

where:

schema=	Name of input schema file to read
outfile=	Name of output file to contain CPDDUTL statements
rpcids=	Name of input file that contains RPC ID definitions
snmnams=	Name of input file that contains SNM schema name definitions

Creating Dictionary Definitions from a Schema File

Creating IT Service Vision table definitions from a SunNet Manager schema file cannot be totally automated because of the need for (sometimes) extensive input from the user. This input is in the form of interpretations of what the schema author had in mind when he or she wrote the schema and provision of a mapping from long, schema variable names to short (less than 8 characters) IT Service Vision variable names. This sometimes results in the user having to edit the schema file and virtually always means the user will have to manually enter the shortened variable names.

To create IT Service Vision dictionary definitions from a schema file, do the following:

1. Define RPC ids

RPC ids are used by SunNet Manager to map numbers to names (of tables). SunNet Manager uses the /etc/rpc file on your system to determine how to log data.

- Locate the /etc/rpc file on the host that runs SunNet Manager. Make a copy of it and edit it to make its format resemble that of the example rpc id file !SASROOT/misc/cpe/csrpc.
- Submit the %CSSNMSCH macro from the SAS PROGRAM EDITOR window to read your schema file (presumably one not already supported by IT Service Vision) and the rpc file.
- Again edit the copy of the rpc file to supply any missing values.
- Loop through these steps until all ids are defined.

2. Map long variable names to short variable names

Schema files use names for tables and metrics. These names are too long for IT Service Vision. You must supply the file that maps long names to short names and point the %CSSNMSCH macro to it.

- Run the %CSSNMSCH macro against your schema file again. It will produce many messages about undefined variable names.

- Create a file that maps long names to short names. Use the messages produced in the preceeding step and the table and variable name definitions in !SASROOT/misc/cpe/cssnmnam as a guide (note that the message produced by %CSSNMSCH in the SAS log are of a form that they can be cut and pasted directly into the file that maps long names to short names. Just type over UNKNOWN with a correct name. There are rules regarding table and variable names for IT Service Vision. They are:
 - User defined table names for IT Service Vision should begin with "U" and have exactly four more alphanumeric characters.
 - Both table and variable names should consist of letters and numbers only (no underscores!).
 - User-defined variables should be 7 or fewer characters long.
- Loop through these steps until all table and variable names have been defined properly.

3. Resolve error messages produced by %CSSNMSCH

A number of things can go wrong with converting a schema file to IT Service Vision dictionary definitions even after the names and rpc ids are resolved. These problems are principally caused by the schema file not being totally syntactically correct. To get around these problems, do the following:

- Run the %CSSNMSCH macro against your schema file again.
- If you get no error messages, continue with step 4.
- If you get error messages, edit the schema file as indicated in the messages. The kinds of error messages to expect are:
 - Malformed BY list. The BY list is called the key in the schema file -- it is used to set the order of observations in the DETAIL data (useful when reporting on the data) and to recognize duplicate observations. Since SunNet Manager does not depend on the BY list being correct (or even present) for tables, it many times is not. Edit the schema file to supply the correct BY list after the "characteristics-k" statement.
 - Too many variables defined in a table. If you get a message about too many variables, change the declaration for the vnam, long, and sort arrays in the %CSSNMSCH macro to something larger and try again.
 - Schema format errors. %CSSNMSCH is more sensitive to some deviations from schema format rules than SNM is. If errors of this type occur, %CSSNMSCH issues messages on the SAS log that are usually self-explanatory if you examine the schema file carefully.
 - Unknown BY-group variable. Until all variable long names are mapped to short names (see step 2 above), this message will appear for the variables in the BY list. Once all long-to-short names have been mapped, this message probably means that the BY-list variable is misspelled in the schema file. A common misspelling involves case mismatch: "-k ifIndex" sometimes is in the BY list definition when "IfIndex" is in the variable definition. Correct the BY list spelling and rerun %CSSNMSCH.
 - Sometimes variations from standard syntax (such as unbalanced parentheses or quotes, or certain special characters) will confound %CSSNMSCH and error messages will be confusing. The easiest way to solve these problems is to examine the schema file for anomalies around the point the error messages occur.
- Loop through these steps until messages have been resolved.

When all messages are resolved, %CSSNMSCH will create, in the specified OUTFILE, the %CPDDUTL statements to define the tables and variables for the schema file.

4. Define the tables and examine/correct the definitions

Despite the best efforts of %CSSNMSCH, some schema files do not convert perfectly to IT Service Vision dictionary definitions.

- Run %CPDDUTL on the dictionary update statements created above to define the new entries in your PDB (a new PDB presumably created just for this purpose).
- Examine the definitions using the IT Service Vision interactive interface and compare them to the schema files. Non-standard constructs or special characters can make %CSSNMSCH parse the schema file incorrectly. Either update the schema file to clarify the variable definition or remove the construct that is confusing %CSSNMSCH.
- Loop through these steps until the definitions are right. If you changed the schema file to correct the problem, loop back through step 3.

5. Use the definitions and correct if necessary.

The ultimate test of a table definition created from a schema file is to use the table.

- Run %CPDDUTL on the dictionary update statements created above to define the new entries in your PDB.
- Test the new definitions by running %CSPROCES on the values of the metrics as logged by SunNet Manager. Modify the %CPDDUTL control statements or the schema file as necessary to get the table(s) the way you want them.
- Loop through these steps until the definitions are right. If you changed the schema file to correct the problem, loop back through step 3.

6. Keep your changes. (optional)

- If you want to make your new table definitions available generally at your site, use the %CPDDUTL INSTALL TABLE control statement. For more details about this statement, see the Macro Reference documentation for IT Service Vision.

EXAMPLE

```
* Allocate IT Service Vision libraries and the PDB;
%CPSTART( mode=batch,
          pdb=/u/testpdb,
          access=write );

* Create CPDDUTL control statements from your own schema file in /u/snm.ddutl;
%CSSNMSCH( schema=/usr/snm/schema/toaster.schema,
          rpcids=!SASROOT/misc/cpe/csrpc,
          snmnams=/u/nams, /* Format like !SASROOT/misc/cpe/cssnmnam */
          outfile=/u/snm.ddutl );

* Create table and variable definitions from the CPDDUTL control statements;
%CPDDUTL( filename='/u/snm.ddutl',
          list=yes );

* Process data from the new schema file;
%CSPROCES( /usr/snm/logfile,
          utabl,
          collectr=SUNETMGR );
```